Projektinhallinnan työkalut osana yrityksen liiketoimintaa

Antti Kantola

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
Pro gradu -tutkielma
Ohjaaja: Timo Poranen
Syyskuu 2013

Tampereen yliopisto
Informaatiotieteiden yksikkö
Antti Kantola: Projektinhallinnan työkalut osana yrityksen liiketoimintaa
Pro gradu -tutkielma, 57 sivua
Syyskuu 2013

Tiivistelmä

Tässä työssä tutkin projektinhallintasovellusten valintaprosessia ja käyttöä kasvuyrityksessä. Tavoitteena on löytää keskeiset aktiviteetit, joita onnistuneen projektin läpivienti edellyttää. Lisäksi tutkin, minkälaisia projektinhallintasovelluksia näiden aktiviteettien tukemiseen ja hallintaan on tarjolla ja mitä haasteita yritykset kohtaavat sovelluksia valitessaan. Etenkin pilvipalveluna toimivien projektinhallintasovellusten määrä on kasvanut viime vuosina merkittävästi ja oikean sovelluksen löytäminen voi olla haastavaa. Osa yrityksistä on päätynyt tekemään projektinhallintasovellluksensa itse sisäisenä työnä, näin tapahtui myös esimerkkiyrityksessä. Toteutin sovelluksen ja tutkin projektin onnistumista mm. teemahaastattelulla.

Sisällys

1 Johdanto

Sekä projektinhallintasovellusten valikoima että yritysten tarve projektiensa tehostamiseen ovat kasvaneet viime vuosina huomattavasti. Kasvu on ollut suurinta pilvipalveluiden valikoimassa joiden etuna on niiden nopea käyttöönotto. Ne eivät myöskään edellytä välttämättä oman palvelimen hankkimista. Lisäksi valmiissa ratkaisuissa ylläpidosta huolehtii kolmas osapuoli.

Yrityskentällä suuri ongelma on yksittäisen yrityksen toimintaa ja projektitarpeiden järkeistämistä palvelevan sovelluksen valinta. Jo yritysten laaja toimialakohtainen kirjo asettaa haasteensa valinnalle ja yhdenkin toimialan sisällä on lukuisia eri tapoja hallinnoida projekteja. Potentiaalisten sovellusten kartoittaminen vaatii paljon työtä, eikä siihen aina ole kasvuyrityksissä riittävää osaamista. Kuitenkin tämä työ on välttämätöntä, sillä yritysten toimintatapojen muuttaminen ilman ammattitaitoa valitun ohjelmiston toimintojen mukaiseksi ei ole yleensä järkevä vaihtoehto.

Perusongelma sovelluksen valinnassa on, että erittäin monien sovellusten liiketoimintaa tukevat ominaisuudet ovat suurilta osin puutteellisia jo yleisestikin yritysten tarvitsemien ominaisuuksien osalta. Sovelluksen käyttötarkoitus ja ohjelmiston rajoitteet on otettava valinnassa luonnollisesti huomioon. Rajoitteiden ja käyttötarkoituksen ymmärtämiseksi on tärkeää ymmärtää ohjelmistokehityksen ja projektinhallinnan vaiheet ja piirteet.

Vaatimusmäärittelyllä ja vaatimusten hallinnalla on suuri rooli projektin onnistumisessa. Toteutuksen suunnittelu etukäteen säästää merkittävästi työaikaa. Yleisten projektinhallintamenetelmien ja niitä yhdistävien piirteiden tuntemus edesauttaa projektinhallinnan ja ohjelmistokehityksen vaiheiden kartoittamista.

Yrityksen toiminnan tunteminen on oleellinen onnistumisen lähtökohta, ja projektinhallintasovellusten liiketoimintaa tukevat ominaisuudet ovat välttämättömiä. Usein niiltä halutaan monipuolisuutta. Samoin tuotantoa on pystyttävä seuraamaan tarkasti, jotta kasvun mahdollisesti tuomiin ongelmiin pystytään reagoimaan. Kasvuyrityksessä kaikki tarpeet eivät ole sovelluksen valinta- tai käyttöönottohetkellä tiedossa, joten sovelluksen on oltava tarpeeksi joustava.

Valmiissa projektinhallintasovelluksissa on sekä hyviä, että huonoja puolia kasvuyrityksen tarpeita ajatellen. Niinpä oman sovelluksen rakentaminen on usein

varteenotettava vaihtoehto.

Tutkin, minkälaisia etuja sisäisenä työnä toteutetulla projektinhallintasovelluksella voidaan saavuttaa. Käyn myös läpi ohjelmistokehityksen vaiheet yleisellä tasolla. Kolmas luku sisältää lyhyen johdannon projektinhallintaan ja muutaman esimerkin ketteristä projektinhallintamenetelmistä. Neljännessä luvussa esittelen nykyaikaisia projektinhallintasovelluksia ja tutkin, miten niiden ominaisuudet tukevat edellisissä luvuissa esitettyjä aktiviteetteja. Viidennessä luvussa esittelen tapaustutkimuksen projektinhallintasovellusten käytön kehityksestä Mediasignal Communications Oy:ssä (nyk. Mediasignal Group). Kuvaan, miten ja miksi yhtiö päätyi nykyiseen projektinhallintakokonaisuuteen - ohjelmoimaan itselleen soveltuvan projektinhallintaohjelmiston kokonaan itse. Pääsin itse seuraamaan prosessia läheltä, sillä toimin ko. sovelluksen toteutuksessa vastaavana ohjelmoijana.

Työni johtaa loppupohdintaan siitä, mitä haasteita yrityksillä on projektinhallintaohjelmistojen valinnassa, mikäli niillä halutaan saavuttaa hyötyjä projektien hallinnassa. Lisäksi pohdin, miten ohjelmistoja kehittävien toimijoiden lähestymistapa palvelee yritysten tarpeita.

2 Ohjelmistokehityksen lähtökohdat yleisesti

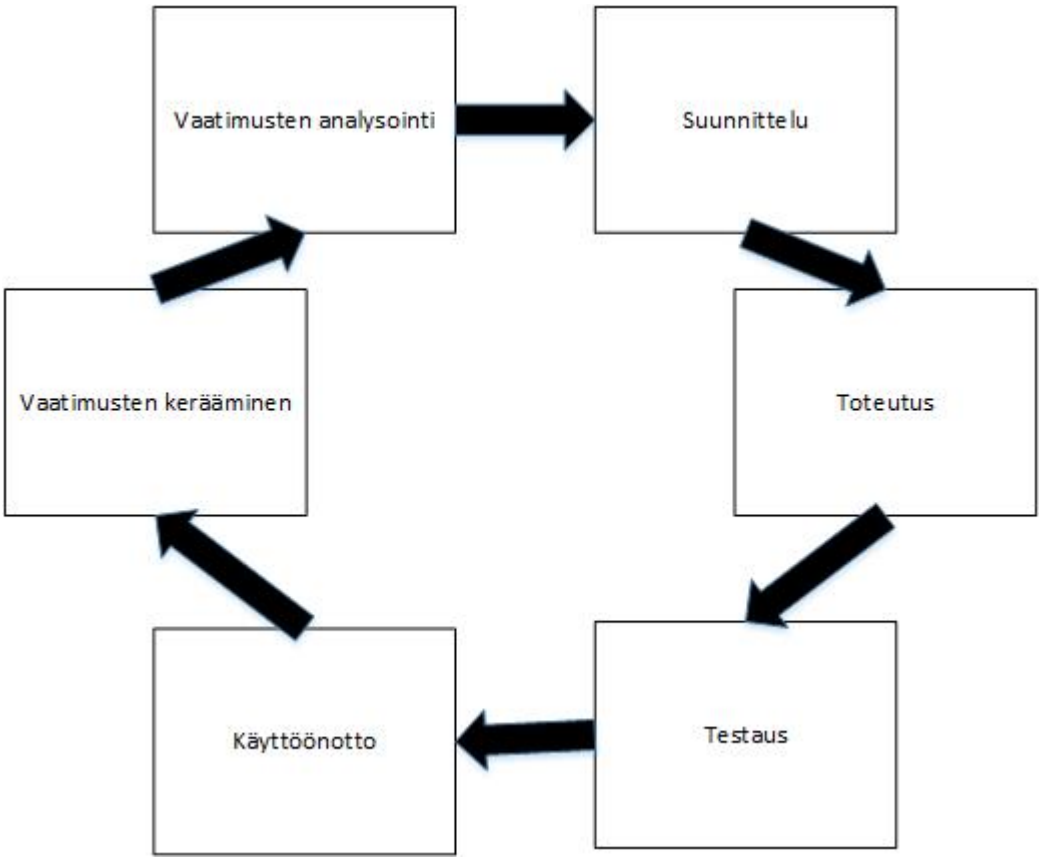
Projekti on ainutkertainen, etukäteen suunniteltu työ, jolla on alku ja loppu. Projektilla on aina jokin ennaltamääritelty tavoite [Chatfield, 2007]. Projekti sisältää tehtäviä, jotka eivät ole rutiininomaisia. Tavanomaisen työn ja projektin erottaa siitä, että projekti vaatii suunnittelua. Lisäksi projektin lopputuloksen hyödyntäjänä on usein joku muu kuin sen toteuttaja. Projekti vaatii usein monen eri alan ammattilaisen asiantuntemusta. Projekti vaatii suunnittelua, mutta jos projektin sovellusalue on tuntematon, saattaa kokeiluun perustuva työskentely olla tehokkaampaa kuin suunnittelu. Tällöin työ on enemmänkin tutkimus- kuin projektiluonteista.

Ohjelmistoprojektien erona muihin teollisen alan projekteihin on toteutuksen tilanteen hahmottamisen vaikeus. Esimerkiksi rakennusalan projekteissa tilanteen arvioiminen on IT-sektoria yksinkertaisempaa johtuen työn konkreettisesta muodosta. Lisäksi projektin työvaiheita ei voida iteroida samalla tavalla kuin ohjelmistokehityksessä [Sommerville, 2011]. Ohjelmistokehityksessä työvaiheet muodostavat iteratiivisen kokonaisuuden, kuten kuvassa 1 on esitetty.

2.1 Vaatimusmäärittely

Ohjelmistokehityksessä vaatimus tarkoittaa järjestelmän käyttäjän edellyttämää ominaisuutta tietyn ongelman ratkaisemiseksi tai päämäärän saavuttamiseksi. Toisaalta vaatimus voi olla jonkin laillisen rajoitteen noudattaminen [IEEE, 1990]. Järjestelmälle asetetut vaatimukset määrittelevät, mitä järjestelmällä on pystyttävä tekemään ja minkälaisessa ympäristössä. Vaatimuksia voidaan määritellä eri tasoilla. Yleisen tason vaatimukset kohdistuvat koko järjestelmään ja sisältävät tyypillisesti asiakkaan perimmäisen tarpeen tai ongelman, joka on motivoinut aloittamaan projektin. Vaatimusmäärittely ja vaatimusten hallinta kestää koko projektin ajan. Vaatimusmäärittelyn onnistuminen on kriittinen vaihe projektin elinkaaressa [Abran & Moore, 2004].

Jokainen vaatimus täytyy olla jäljitettävissä sen alkuperäiseen liiketoiminnan tavoitteeseen. Jäljittämällä vaatimus sen alkuperään voidaan tarkistaa, ettei vaatimus ole muuttunut, ja varmistamaan, että se on testattu ja tukee projektin tavoitteita [Jama Software, 2011].



Kuva 1 Ohjelmistokehityksen vaiheet.

2.1.1 Projektin osakkaat

Projektin osakkailla tarkoitetaan niitä henkilöitä ja organisaatioita, joihin projekti tai sen lopputulos vaikuttaa. Osakkaat asettavat vaatimuksia projektille. Projektin osakkaita voivat olla esimerkiksi projektin rahoittava asiakas, loppukäyttäjä, joka lopulta tulee käyttämään valmista järjestelmää, sovellusalueen asiantuntija, joka saattaa olla joku projektia varten palkattu asiantuntija tai asiakasyrityksen työntekijä, vaatimusmäärittelijä, projektipäällikkö, ohjelmistoarkkitehti sekä järjestelmän toteuttavat ohjelmoijat. Vaatimusmäärittelyyn kuuluu yhtenä aktiviteettinä näiden osakkaiden tunnistaminen [Kotonya & Sommerville, 1998].

Eri osakkailla on eri intressit toteutettavan järjestelmän suhteen. Asiakkaan tavoitteena on saada tuottoa sijoitukselleen joko myyntituottoina tai liiketoiminnan tehostamisen kautta. Loppukäyttäjä haluaa ratkaista järjestelmän avulla jonkin tietyn ongelman. Vaatimusmäärittelijä taas pyrkii toteuttamaan edellämainittujen

vaatimukset. Projektipäällikkö haluaa toteuttaa vaatimukset mahdollisimman tehokkaasti. Ohjelmistoarkkitehdin vaatimukset liittyvät järjestelmän tekniseen kokonaisuuteen. Ohjelmoijien asettamat vaatimukset voivat koskea esimerkiksi lähdekoodin luotettavuutta ja muunneltavuutta. Yhteensä vaatimuksia kertyy kattavassa määrittelyssä satoja tai jopa tuhansia [Jama Software, 2011].

2.1.2 Ei-toiminnalliset vaatimukset

Ei-toiminnalliset vaatimukset ovat vaatimuksia, jotka eivät välittömästi liity järjestelmän toiminnallisuuteen. Sen sijaan ne asettavat rajoitteita ja ehtoja toteutettavalle järjestelmälle ja projektille. Ei-toiminnalliset vaatimukset ovat tavallisesti lähtöisin järjestelmän käyttöympäristöstä. Järjestelmän suunnittelusta alkaen on huomioitava ympäristön asettamat vaatimukset ja rajoitteet. Kriittisissä järjestelmissä, kuten sairaalassa elintoimintoja mittaavat järjestelmät tai lennonjohdon järjestelmissä, käyttökohteen ja ympäristön kriteereitä ei voida jättää huomiotta [Sommerville, 2011].

Ei-toiminnalliset vaatimukset voidaan ryhmitellä kolmeen ryhmään, joista jokainen sisältää vielä omat alaryhmänsä. Pääryhmät ovat prosessivaatimukset, tuotevaatimukset ja ulkopuoliset vaatimukset. Ryhmät on esitetty kuvassa 2.

Kuva 2 Ei-toiminnalliset vaatimukset.

Prosessivaatimukset asettavat vaatimuksia ja rajoitteita järjestelmän toteutukselle. Tällaisia vaatimuksia esiintyy enemmän suurissa kuin pienissä yrityksissä.

Suurilla yrityksillä saattaa olla hyvinkin tarkkoja standardeja, joiden mukaisesti järjestelmä tulee toteuttaa. Prosessivaatimukset jakaantuvat edelleen kolmeen ryhmään, toimitus-, toteutus-, ja standardivaatimuksiin. Standardivaatimus voi esimerkiksi olla jonkun tietyn projektinhallintamenetelmän käyttö ja toteutukseen liittyvä prosessivaatimus voi edellyttää tiettyjen ohjelmointityökalujen käyttöä.

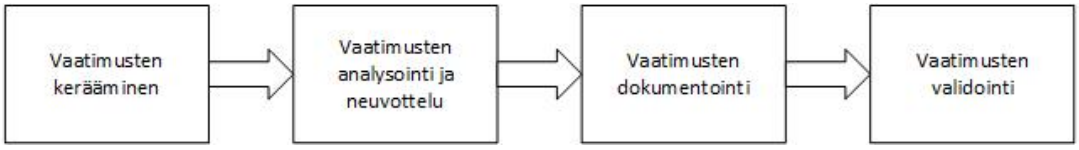
Tuotevaatimukset käsittävät kaikki tuotteeseen liittyvät ei-toiminnalliset vaatimukset. Ryhmiä on neljä, joista yksi jakaantuu vielä kahteen alaryhmään: käytettävyysvaatimukset, luotettavuusvaatimukset, turvallisuusvaatimukset ja tehokkuusvaatimukset. Tehokkuusvaatimukset jakautuvat vielä kahteen ryhmään, jotka ovat suorituskykyvaatimukset ja kapasiteettivaatimukset. Tuotevaatimuksille on tyypillistä niiden väliset ristiriidat. Esimerkiksi jokin tietoturvaan liittyvä vaatimus saattaa aiheuttaa suorituskyvyn heikentymistä.

Ulkoiset vaatimukset ovat laillisia ja taloudellisista rajoitteita tai yhteensopivuusvaatimuksia. Vaatimukset kohdistuvat joko itse tuotteeseen tai prosessiin [Kotonya & Sommerville, 1998].

2.1.3 Vaatimusmäärittelyn vaiheet

Järjestelmän suunnittelijat työskentelevät asiakkaan ja loppukäyttäjän kanssa vaatimusten keräämiseksi. Pelkkä kysyminen tai haastattelu harvoin riittää osakkaiden todellisten tarpeiden ja ongelmien paljastamiseksi.

Vaatimusmäärittelyn voi vaiheistaa monella eri tapaa, mutta kaikista hyvistä vaatimusmäärittelyprosesseista pitäisi löytyä ne neljä vaihetta, jotka on esitetty kuvassa 3.



Kuva 3 Vaatimusmäärittelyn vaiheet.

Ensimmäisessä vaiheessa kartoitetaan yleisellä tasolla projektin rahoittavan yrityksen liiketoiminnan tavoitteet ja toteutettavan järjestelmän tavoiteltavat vaikutukset. Samalla voidaan jo määrittää projektin budjetti ja aikataulu. Toisessa

vaiheessa hankitaan mahdollisimman paljon taustatietoa kaikesta projektin aihealueeseen liittyvästä. Kolmannessa vaiheessa hankittu tieto organisoidaan. Organisoinnin lähtökohtana voidaan käyttää projektin osakkaiden rooleja. Organisointiin kuuluu myös osakkaiden tavoitteiden priorisointi. Neljännessä vaiheessa projektin osakkaiden kanssa tehdään yhteistyötä vaatimusten tunnistamiseksi haastattelemalla ja tarkkailemalla heitä. Prosessin aikana kartoitetaan mahdollisimman kattavasti asiakkaan, loppukäyttäjän ja muiden projektin osallisten tarpeet. Tarpeiden kartoitus on hankalaa, sillä harvoin asiakas osaa suoraan kertoa tarkasti ja kokonaisvaltaisesti tarpeitansa. Kartoituksen apuna käytetään tyypillisesti erilaisia haastatteluja, kyselyitä, nykyisten työskentelytapojen havainnointia ja prototyyppejä [Kotonya & Sommerville, 1998].

Vaatimusten kerääminen

Vaatimuksia kerätessä tulee ottaa huomioon kaikki osapuolet, joihin projektin toteuttaminen tai lopputulos vaikuttavat. Tyypillisesti näitä tahoja ovat ainakin järjestelmän loppukäyttäjät, asiakas sekä projektiryhmä [Rowel & Alfeche, 1997]. Vaatimusten kerääminen ihmisiltä on työlästä, sillä ihmiset eivät pysty täydellisesti määrittelemään tarpeitaan monimutkaiselle järjestelmälle. Projektiin osallistuvien henkilöiden lisäksi vaatimuksia asettavat mahdollisesti ennestään olemassa oleva järjestelmä, organisaation standardit ja käytännöt sekä ulkoiset säännöt.

Ennestään olemassa oleva järjestelmä saattaa asettaa rajoituksia esimerkiksi mahdollisten rajapintojen suhteen. Rajapintoja voi olla jo olemassa kolmansien osapuolten järjestelmiin, eikä näistä rajapinnoista ole välttämättä tarkoitus projektin yhteydessä luopua. Samoin nykyisessä järjestelmässä saattaa olla hyvin toimiva laitteisto tai käyttöliittymä, jotka myöskin halutaan säilyttää. Projektiin kohdistuvat organisaation vaatimukset voivat tarkoittaa esimerkiksi tietyn ohjelmointikielen tai palvelinalustan käyttöä.

Ulkoisilla säännöillä tarkoitetaan tässä yhteydessä lähinnä lainsäädäntöä, jotka asettavat rajoituksia tai lisävaatimuksia. Esimerkkinä henkilötietolaki, joka määrittelee, miten henkilötietoja saadaan järjestelmän rekisteriin tallentaa ja mihin niitä saadaan käyttää. Edellä mainittujen lisäksi vaatimuksia asettavat sovellusalueelle syntyneet yleiset käytännöt, kuten vaikkapa kirjastossa kirjojen yksilöiminen ISBN-tunnisteilla. Aihealueen tuntemus auttaa ymmärtämään käyttäjien vaatimuksia [Kotonya & Sommerville, 1998].

Vaatimusten analysointi

Kun vaatimuksia on kerätty, kootut vaatimukset analysoidaan. Analysoinnin tuloksena päätetään, hyväksytäänkö vaatimusta toteutettavaksi. Analysointi suoritetaan projektin jäsenten välisessä keskustelussa. On tavallista, että eri jäsenten väliset näkemyserot aiheuttavat konflikteja ja kaikkia vaatimuksia ei voida hyväksyä. Hyväksytyt vaatimukset dokumentoidaan tarkasti vaatimusmäärittelydokumenttiin. Lopuksi kaikki dokumentoidut vaatimukset käydään vielä läpi virheiden ja epäjohdonmukaisuuksien varalta. Vaatimukset analysoidaan yhdessä projektin osakkaiden kanssa epämääräisyyden ja mahdollisten ristiriitojen varalta. Analysointia tehdään koko vaatimusmäärittelyprosessin ajan.

Erikseen vaatimusten analysointia ja vaatimusten ristiriitoja varten järjestetyt kokoukset ovat yksi tapa neuvotella ja ratkaista ongelmia. Analysoinnin yhteydessä suoritetaan vaatimusten priorisointia. Priorisoinnilla pyritään tunnistamaan ne vaatimukset, jotka ovat kriittisiä liiketoimintatavoitteiden kannalta. Vastaavasti on tärkeää tunnistaa ne vaatimukset, jotka eivät käytännössä tuo lisäarvoa järjestelmälle. Edellämainittujen ääripäiden lisäksi luokitellaan loput vaatimukset keskitason ja matalan tason vaatimuksiin. Vaatimusten tärkeyden ohella priorisointiin vaikuttaa monesti käytännössä asiakkaan taloudelliset resurssit projektin toteuttamiselle.

Vaatimusten dokumentointi

Vaatimusmäärittelydokumentti voi olla muodoltaan vapaamuotoista tekstiä sisältäen erilaisia käyttöskenaarioita ja -tapauksia. Alkuperäiset vaatimukset, joista dokumentin sisältö johdetaan, voivat sen sijaan olla kaikkea tarkkojen lakitekstien ja ruutupaperille käsin piirrettyjen kuvien tai kuvioiden väliltä [Jama Software, 2011]. Dokumentin ensisijainen tarkoitus on esitellä vaatimusmäärittelyn tulokset asiakkaalle, jotta asiakas voi arvioida ja hyväksyä vaatimukset. Asiakkaan lisäksi dokumentti palvelee myös muita projektin osakkaita. Dokumentista pitäisi ilmetä järjestelmän toiminnallisuus, rajaus ja käyttöympäristö. Rajauksella tarkoitetaan kriteereitä ja rajoitteita, jotka vaikuttavat sekä järjestelmän kehitykseen että käyttöön. Rajoitteita voivat olla esimerkiksi projektin toteuttavan organisaation standardit. Käyttöympäristöllä taas tarkoitetaan niitä olosuhteita, joissa järjestelmä toimii. Käyttöympäristön muodostaa laitteisto- ja ohjelmistorajapinnat [Smith, 2008].

Vaatimusten hallinta

Vaatimusten hallinta on läpi projekin kestävä prosessi, jonka tarvoittena on organisoida dokumentoidut vaatimukset siten, että voidaan helposti nähdä, mitkä vaatimukset on jo täytetty ja mitkä ovat täyttämättä [Delgadillo & Gotel, 2007].

Eräs ketteriin ohjelmistokehitysmenetelmiin sopiva menetelmä on StoryWall-konsepti. Konsepti perustuu fyysisiin kortteihin, joihin on kirjoitetaan järjestelmän käyttöön liittyvä tarina. Kortit kootaan seinälle, jossa ylläpidetään kirjaa täytetyistä ja täyttämättömistä tarinoista. Seinällä olevia kortteja voidaan ryhmitellä tai järjestää niiden prioriteetin mukaan [Delgadillo & Gotel, 2007]. Menetelmää voi soveltaa käyttäen oikeita kortteja ja oikeaa seinää, mutta tarjolla on myös StoryWall-web-ohjelma [StoryWall, 2011]. Ohjelman käyttö on yhtä yksinkertaista kuin perinteistenkin välineiden. Virtuaaliselle seinälle luodaan tarvittava määrä sarakkeita, joihin kortteja järjestetään. Kortteja voi ryhmitellä ja merkitä käyttäen valmiita ikoneita ja värejä.

Vaatimusten hallinta edellyttää kaikkien projektin osakkaiden sisäistäneen projektin tarkoituksen ja tavoitteet. Projektin asiakkaan tavoitteet asettavat kontekstin järjestelmän vaatimuksille ja koko projektille. Kontekstin asettaminen auttaa projektiryhmää ymmärtämään vaatimuksia, jolloin vaatimusten muutokset ovat paremmin ennustettavissa [Jama Software, 2011].

Yhtenä keskeisenä ongelmana vaatimusten hallinnassa on vaatimusten jatkuva muuttuminen kesken projektin. Vaatimusten muuttuessa vaatimukset täytyy tarkistaa muuttuneiden suhteiden, prioriteettien tai riippuvuuksien varalta.

2.1.4 Ongelmat

Projektin rajauksesta saattaa olla ristiriitaisia käsityksiä osakkaiden välillä. Seurauksena kasaantuu tarpeettomia yksityiskohtia, jotka lisäävät vaatimusten epämääräisyyttä ja sekoittavat projektin tavoitteita. Osakkailla saattaa olla myös heikko käsitys teknisistä rajoitteista tai heidän toimialatuntemus ei ole riittävä autenttisten tarpeiden ilmaisemiseksi. Toisaalta vahvasti kohdealueen hallitsevat, ja sen kanssa päivittäin toimivat henkilöt saattavat virheellisesti olettaa joidenkin vaatimusten olevan ilmeisiä vaatimusten kerääjille ja siksi jättää kriittisiä yksityiskohtia paljastamatta. Entisestään tilannetta hankaloittaa vaatimusten epävakaus ja muutokset projektin elinkaaren aikana [Christel & Kang, 1992].

2.2 Suunnittelu

Vaatimusmäärittelyn pohjalta tehtävän suunnittelun tavoitteena on muodostaa kuvaus järjestelmän rakenteesta, tietomalleista, käyttöympäristöstä, kuten myös sisäisistä ja ulkoisista rajapinnoista. Suunnittelu tehdään iteratiivisesti tarkentamalla kuvausta tarkistuspisteissä tehtyjen päätösten perusteella [Sommerville, 2011].

Suunnittelun aloittamiseksi tarvitaan tyypillisesti kuvaus sovellusalustasta, aikaisemmin kootuista vaatimuksista ja sovellusalueen tietomallista. Kun tarvittava pohjamateriaali on saatu kasaan, voidaan aloittaa arkkitehtuurin, rajapintojen, komponenttien ja tietokannan suunnittelu. Kaikkia aktiviteetteja voidaan edistää rinnakkaisesti ja ne tukevat toisiaan. Järjestelmästä riippuen aktiviteetit voivat vaihdella. Esimerkiksi useilla reaaliaikaisilla järjestelmillä ei välttämättä ole lainkaan tietokantaa [Sommerville, 2011].

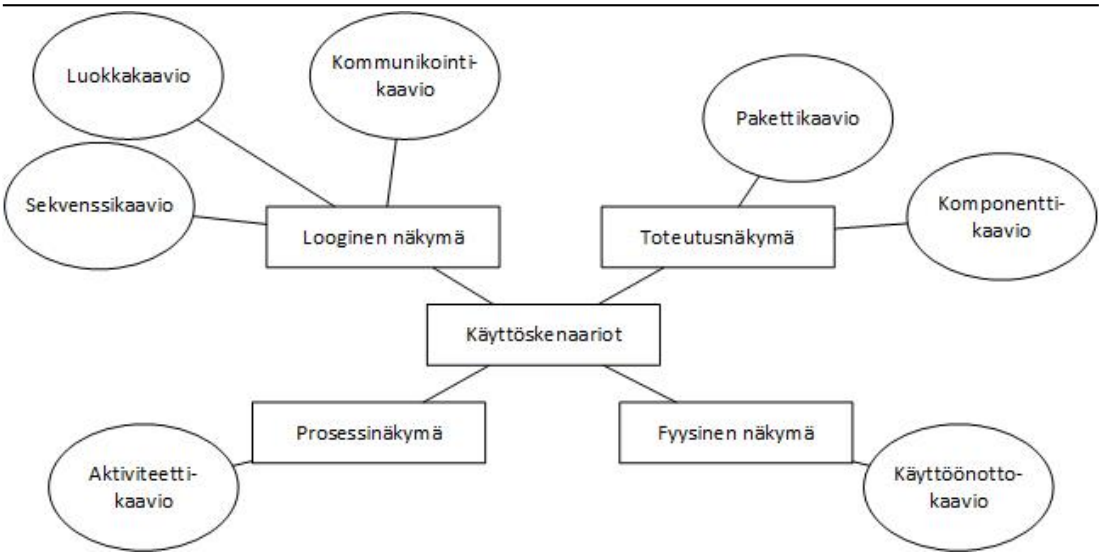
2.2.1 Arkkitehtuurisuunnittelu

Arkkitehtuurisuunnitelman tarkoituksena on kuvata järjestelmän rakenne yleisellä tasolla. Tavoitteena on esitellä ja kuvata keskeisimmät komponentit ja niiden relaatiot sekä roolit järjestelmässä [Sommerville, 2011]. Arkkitehtuurisuunnitelma tarjoaa viitekehyksen ensimmäisille suunnitteluun liittyville päätöksille. Varsinaisten suunnitelmien lisäksi syntyvästä dokumentista tulisi selvitä suunnitteluratkaisujen taustalla vaikuttavat perusteet ja muut syyt kyseisille päätöksille [Babar, 2009]. Käytännössä tämä tarkoittaa tavallisesti viittausta johonkin vaatimusmäärittelydokumentin kohtaan.

Konkreettinen dokumentti arkkitehtuurista helpottaa projektin osakkaiden välistä keskustelua ja vähentää epämääräisyyttä. Toinen arkkitehtuurin dokumentoinnin merkkittävä hyöty on suunnitelmien uudelleenkäyttömahdollisuus yrityksen tulevissa projekteissa [Bass *et al.*, 2012]. Tästä taas seuraa luonnollisesti väärinkäsitysten ja sitä kautta kustannusten väheneminen [SARA Work Group, 2002].

Arkkitehtuuria voidaan katsoa monesta eri näkökulmasta ja jokaisella projektin osakkaalla on omat intressinsä. Dokumentin tulisi palvella kaikkia osapuolia ja näkökulmia.

Yksi tapa jakaa dokumentti osiin on käyttää 4+1 -mallia [Kruchten, 1995]. 4+1 -malli sisältää neljä näkymää, joista jokainen kuvaa järjestelmää eri näkökulmasta. Näkymät koostuvat UML-kaavioista. Eri näkökulmien tarkoitus on kuvata järjestelmän arkkitehtuuria projektin osakkaille kunkin osapuolen intressejä palvelevalla tavalla. Neljä mallia ovat looginen näkymä (logical view), kehitysnäkymä (development view), prosessinäkymä (process view) ja fyysinen näkymä (physical view). Viides osa kyseistä mallia on skenaariot (scenarios) [Kruchten, 1995].



Kuva 4 4+1 -malli.

Looginen näkymä kuvaa järjestelmän loppukäyttäjälle tarjoamat toiminnallisuudet. Käytännössä puhutaan luokkakaavioista, kommunikointikaavioista ja sekvenssikaavioista. Kehitysnäkymä sisältää järjestelmän kehittäjälle tarpeelliset kaaviot, kuten komponentti- ja pakettikaaviot. Prosessinäkymä havainnollistaa järjestelmän prosesseja ja niiden kommunikointia järjestelmän käytön aikana. Prosessinäkymästä pitäisi pystyä lukemaan järjestelmän prosessien yhtäaikaisuutta, hajautuneisuutta, integraattoreita, suorituskykyä ja skaalautuvuutta. Fyysinen näkymä sisältää nimensä mukaisesti kuvaukset järjestelmän laitteistosta ja niiden välisistä suhteista ja riippuvuuksista. Skenaariot eivät ole varsinaisesti näkymiä ja siitä johtuu mallin nimi 4+1. Skenaariot ovat usein tekstimuotoisa kuvauksia järjestelmän todenmukaisista käyttötavoista [Kontio, 2005].

Arkkitehtuurin dokumentointiin tarvittava materiaali syntyy yhteistyössä projektin osakkaiden kanssa. Ohjelmistoarkkitehti neuvottelee muiden projektin osakkaiden kanssa vaatimuksista ja tarvittaessa ratkoo vaatimusmäärittelyn ristiriitoja.

Mikäli johonkin vaatimusmäärittelyn kohtaan löytyy jo aikaisemmin käytetty suunnittelumalli, selvittää ohjelmistoarkkitehti mahdollisuutta uudelleenkäyttää sitä [Kruchten, 2008].

2.2.2 Rajapintasuunnittelu

Ohjelmistoteollisuuden kehityksen myötä toteutettavien ohjelmistojen monimutkaisuus ja vaatimukset kasvavat. Komponentteihin perustuva suunnittelu korostaa sovellusalueen objektien roolia ohjelmistokehityksessä. Tavoitteena on komponenttien täysimittaisen kehittämisen sijaan hankkia uudelleenkäytettäviä ratkaisuja ja siten vähentää toteutustyöhön tarvittavia resursseja [Tran & Liu, 1999]. Mikäli komponentit hankitaan ulkoisista lähteistä, liittyy niiden käyttöön kuitenkin riskejä. Vasta myöhemmin saattaa paljastua, ettei kyseinen komponentti kelpaakaan sille suunniteltuun tarkoitukseen. Sen vuoksi komponentin rajapintojen tarkka määrittely komponentteja kehitettäessä on tärkeää paitsi nykyisen, mutta myös tulevien projektien kannalta.

Rajapinta on kokoelma liittymiä, joiden avulla komponenttia käytetään. Ongelma, jota rajapintasuunnittelulla yritetään täsmentää, on tietyn komponentin näkyvyys muille järjestelmän komponenteille. Määrittelyiden tarkoituksena on tarjota kuvaus järjestelmän suunnittelijoille ja toteuttajille [Bosch, 1997]. Rajapintasuunnitelman määrittelyiden on oltava yksiselitteisiä. Näin komponenttien suunnittelu ja toteutus voidaan jakaa eri ryhmille, jotka voivat työskennellä itsenäisesti rajapintasuunnitelmaa noudattaen [Sommerville, 2011].

2.2.3 Komponentti- ja tietokantasuunnittelu

Rajapintasuunnittelun tuloksena syntyneiden dokumenttien avulla voidaan aloittaa komponenttisuunnittelu. Komponenttisuunnitelma sisältää kuvauksen jokaisesta järjestelmän toiminnallisesta osasta ja määrittelyn sen toimintatavasta. Toteutukseen liittyvien yksityiskohtien, kuten luokkien, määrittely jätetään toteuttajille [Sommerville, 2011].

Tietokantasuunnitelma kuvaa järjestelmän tietorakenteet ja niiden tallennustavan tietokannassa [Sommerville, 2011]. Tietokannan rakenteen määrittely aloitetaan käytännössä jo vaatimusmäärittelyn yhteydessä. Vaatimusmäärittelyn aikana kerätystä materiaalista ilmenee aihepiirin muuttujat, joista tietokannan rakenne

muodostu. Yleisimmät tavat mallintaa tietoa ovat ER-kaaviot ja UML-mallit [Badia, 2011].

2.3 Toteutus ja testaus

Toteutus tehdään aikaisemmin tehtyjä suunnitteludokumentteja noudattaen. Tekniset yksityiskohdat jätetään kuitenkin itse ohjelmoijille ratkaistavaksi. Lisäksi ohjelmoijat voivat itse suunnitella komponenttien toteutusjärjesteyksen. Toteutuksessa pyritään välttämään tarpeetonta monimutkaisuutta ja sen sijaan korostetaan yksinkertaista ja luettavaa lähdekoodia. Monimutkaisuuden välttäminen onnistuu käyttämällä mahdollisimman paljon standardoituja ohjelmointitekniikoita. Lähdekoodin yksinkertaisuus on tärkeä tekijä järjestelmän tulevaisuudessa tehtävän ylläpidon kannalta [Romano *et al.*, 2004].

Ohjelmoijat suorittavat yksikkötestausta ja virheiden etsintää toteutuksen aikana ja varsinainen validointi tehdään vasta toteutuksen jälkeen. Yksikkötestaus tarkoittaa jonkin lähdekoodin osan testaamista tarkkaan valituilla syötteillä [Dorota Huizinga, 2007]. Yksiköksi lasketaan pienin mahdollinen testattavissa oleva lähdekoodin osa. Testattavissa voi olla kokonainen luokka tai mahdollisesti luokan yksittäinen funktio. Joissakin ohjelmointiparadigmoissa, kuten XP:ssä ja Scrumissa, on tavallista kirjoittaa yksikkötestit ennen varsinaista lähdekoodia. Koodi hyväksytään, kun kaikki testit on läpäisty.

Yksikkötestauksella on lukuisia hyötyjä. Ohjelmointivirheet löytyvät aikaisemmin, jolloin niiden korjaaminen on nopeampaa ja helpompaa. Samoin ylläpidettävyys helpottuu, kun yksikkötestien avulla voidaan varmistaa, että päivitetty lähdekoodi toimii edelleen vaaditulla tavalla. Yksikkötestit palvelevat myös lähdekoodin dokumentointia, sillä yksikkötesteistä näkee nopeasti, miten yksikön on tarkoitus toimia. Testaus vaatii kuitenkin aina työtä ja huolellisen testaamisen suorittaminen saattaa vaatia uuden testiluokan ohjelmointia ja joskus sitä ennen jopa luokan suunnittelun.

Järjestelmän validoinnin tarkoitus on varmistaa, että toteutettu järjestelmä tai sen osa toimii käyttäjän vaatimalla tavalla. Validointi on testausta korkealla tasolla. Validointia voidaan tehdä joko yksittäiselle komponentille tai koko järjestelmälle. Toisaalta järjestelmän kokonaisvaltainen validointi on usein tarkoituksenmukaista vain pienille järjestelmille. Suurelle järjestelmälle se saattaa olla liian tehotonta ja epätarkkaa [Sommerville, 2011].

Validointi suoritetaan tarkastamalla järjestelmän toiminnallisuutta ja toimintalogiikkaa alkuperäisiä vaatimuksia vasten. Validointi sekoitetaan usein verifiointiin, joka on kylläkin myös liittyy järjestelmän testaukseen. Validointi arvioi järjestelmän soveltuvuutta alkuperäisen ongelman ratkaisuun ja sitä, täyttääkö se loppukäyttäjän tarpeet. Verifiointi taas puolestaan arvioi järjestelmän toimivuutta ja toteutustapaa [Boehm, 1989].

3 Projektinhallinta

Projektinhallinnalla tarkoitetaan projektin suunnittelua ja organisointia siten, että kaikki projektin päämäärät toteutuvat ennaltamääriteltyjen resurssien ja rajojen puitteissa [Hughes, 2006]. Rajoja asettavat osaltaan esimerkiksi projektin laajuus, käytettävissä oleva aika ja budjetti. Projektinhallinnan tavoitteena on optimoida ja jakaa käytettävissä olevia resursseja mahdollisimman tehokkaasti päämäärien saavuttamiseksi [Project Management Institute, 2009].

Projektinhallintaa on harjoitettu tuhansia vuosia, mutta vasta 500 vuotta sitten organisaatiot alkoivat käyttää systemaattisia projektinhallinnan työkaluja ja tekniikoita monimutkaisten projektien läpiviemiseksi. Projektinhallinta alkoi saavuttaa nykyisen muotonsa 1950-luvulla, kun Yhdysvaltain laivasto käynnisti Polaris-projektinsa [Sapolsky, 2004]. Myöhemmin Yhdysvaltain ilmailu- ja avaruushallinto NASA sekä puolustusministeriö käyttivät vastaavia projektinhallintamenetelmiä suurissa hankkeissaan. Ohjelmistokehityksen alalla sivistyneitä projektintamenetelmiä alettiin omaksua 1980-luvun alussa ja 1990-luvun alkuun mennessä projektinhallintamenetelmien tekniikat olivat löytäneet tiensä valtaosaan eri alan yrityksiä ja organisaatioita [Kwak, 2003].

3.1 PRINCE2

PRINCE2 (PRojects IN Controlled Environments) on Isossa-Britanniassa laajasti käytetty strukturoitu lähestymistapa projektinhallintaan. Menetelmän avainpiirteitä ovat [ILX Group, 2013b]

- liiketoiminnan tavoitteiden korostaminen,
- projektiryhmän organisaation rakenteen määrittäminen,
- tuotekeskeinen suunnittelu,
- projektin vaiheistamisen keskeisyys ja
- hallinnan mukautuvuus projektin koon mukaan.

PRINCE2 on suunnattu yrityksille ja yhteisöille, joilla on useita eri kokoisia projekteja käynnissä samaan aikaan. Tavoitteena on soveltaa samaa projektinhallintakehystä kaikkiin projekteihin koosta riippumatta.

Suunnitteluvaiheessa määritellään ja analysoidaan järjestelmän osat ja niiden toteuttamiseen vaadittavat työosiot. Jokainen työosio käydään erikseen läpi ja määritellään sille työaika-arvio. Kun työaika-arviot on saatu jokaiselle työosiolle, voidaan niiden pohjalta suunnitella koko projektin aikataulu. Aikataulua suunniteltaessa huomioidaan ja analysoidaan myös projektin riskit. Havaituista riskeistä voi aiheutua uusia työosioita [Cotterell & Hughes, 2009].

Ennen varsinaista suunnittelua tehdään suunnitelman suunittelu. Suunnitelman suunnittelu tarkoittaa suunnitelman sisällön ja vaaditun tarkkuuden määrittelyä. Tämä vaihe edesauttaa menetelmän mukautuvuutta eri kokoisiin projekteihin. Suunnittelussa vaaditun tarkkuuden taso riippuu toteutettavan järjestelmän kriittisyydestä. Esimerkiksi liiketoiminnalle erityisen kriittinen järjestelmä vaatii tarkemman tason suunnittelua kuin pelkkä taloudellisen tuoton tehostamiseen tarkoitettu järjestelmä [Cotterell & Hughes, 2009].

3.1.1 Roolit ja vaiheistus

Jokaisella projektin osakkaalla on projektissa jokin rooli. Sama henkilö voi olla useammassa roolissa ja vastaavasti samassa roolissa voi olla useampi henkilö riippuen projektiorganisaation koosta. Johtoryhmä sisältää yleensä ainakin asiakkaan, loppukäyttäjän ja toimittajan edustajat [ILX Group, 2013a]. Toimittaja vastaa siitä, että asiakkaan vaatimukset tulevat täytetyiksi. Projektilla on projektipäällikkö, joka kontrolloi projektin resursseja ja vastaa projektin edistymisestä projektin johtoryhmälle. Isoissa projekteissa projektipäälliköllä saattaa olla käytettävissään työryhmän vetäjiä. Projektisihteeri hallitsee ja organisoi projektin dataa, kuten suunnitteludokumenttien, ohjelmistokomponenttien ja muiden projektin tuotoksien eri versioita.

Projektin osat jaetaan loogisesti yksittäisesti hallittaviksi kokonaisuuksiksi. Projektipäälliköllä on johtoryhmältä valtuudet yhden vaiheen suorittamiseen kerrallaan. Vasta kun johtoryhmä on hyväksynyt seuraavan vaiheen suunnitelmat, voidaan siirtyä eteenpäin. Jokaisen vaiheen välissä johtoryhmä tarkastaa projektin resurssit, tilanteen ja että projekti on edelleen tarkoituksenmukainen asiakkaan liiketoiminnalle. Projektin suunnitteluvaiheessa projektin vaiheet suunnitellaan hyvin

yleisellä tasolla. Ennen jokaisen vaiheen aloittamista suunnitelmaa tarkennetaan sen vaiheen osalta [Cotterell & Hughes, 2009].

Projektin ohjaaminen sisältää toimenpiteitä projektin eri vaiheiden valtuuttamiseksi. Projektin ohjaaminen kuuluu ensisijaisesti projektin johtoryhmälle, mutta myös projektipäällikkö voi vaikuttaa päätöksiin. Esimerkkejä prosessin toimenpiteistä ovat suunnittelutyön valtuuttaminen, suunnitelmien hyväksyminen ja projektin käynnistämisen valtuuttaminen, projektin vaiheiden hyväksyminen ja projektin päättäminen [Cotterell & Hughes, 2009].

3.1.2 Projektin perustaminen ja aloittaminen

Asiakkaan organisaatio perustaa projektin. Asiakkaalla on todennäköisesti useita potentiaalisia projekteja, jotka tehostaisivat yrityksen liiketoimintaa. Projektin perustaminen ei vielä sisällä projektin suunnittelua. Sen sijaan se sisältää oikeiden henkilöiden ja dokumenttien sekä muiden materiaalien kokoamista [Cotterell & Hughes, 2009].

Projektin perustaminen ja käynnistäminen on jaettu loogisesti eri prosesseiksi. Kun projektin johtoryhmä perustamisen jälkeen toteaa, että projektin jatkaminen on kannattavaa, voi projektipäällikkö käynnistää projektin. Käynnistäminen tarkoittaa käytännössä projektipäällikön tekemää tarkempaa suunnittelua. Perustamisvaiheessa kerätyn materiaalin perusteella laatii projektipäällikkö projektisuunnitelman [Cotterell & Hughes, 2009]. Projektisuunnitelma sisältää yleensä

- toteutettavan järjestelmän keskeisimmät osat,
- projektin työosiot,
- riskienhallintasuunnitelman,
- työmääräarviot,
- aikataulun ja
- seurantapisteet.

Johtoryhmän hyväksyttäväksi laaditaan prosessin aikana koottu projektin käynnistämisdokumentti [Cotterell & Hughes, 2009].

3.1.3 Kontrollointi ja rajoitteet

Käynnistämisen jälkeen voidaan aloittaa työosioiden tuotanto. Tuotannon organisointi ja kontrollointi kuuluu projektipäällikölle ja mahdolliselle tiiminvetäjälle. Projektipäällikön tehtäviin tuotannon aikana kuuluu työpakettien hyväksyntä, nykytilanteen ja edistymisen arviointi, ongelmien ratkominen ja johtoryhmälle raportointi. Ongelmia, kuten uusia vaatimuksia, teknisiä ongelmia ja laillisia rajoitteita ilmenee vääjäämättä ja suurin osa projektipäällikön ajasta kuluu niiden ratkomiseen. Kaikki tällaiset tapaukset tulisi kirjoittaa muistiin myöhempiä projekteja varten. Ongelmien ratkomisen lisäksi projektipäällikkö seuraa projektin etenemistä [Cotterell & Hughes, 2009].

Projektin työosioiden tarkat suunnitelmat laaditaan vasta, kun työosion aloitusajankohta lähestyy. Työosion suunnitelmaa laadittaessa saattaa ilmetä uusia vaatimuksia ja uusia työosioita, joita projektin toteuttaminen vaatii. Uudet tarpeet tarkoittavat lähes poikkeuksetta lisäkustannuksia ja koko projektin valmistumisajankohdan viivästymistä. Uudet tarpeet täytyy käsitellä projektinhallintasuunnitelmassa. Jos muutokset ovat suuria, täytyy projektipäällikön laatia poikkeusraportti johtoryhmälle [Cotterell & Hughes, 2009].

Projekti päätetään, kun kaikki ongelmat on ratkaistu tai niiden ratkaiseminen on määritelty jatkokehitykseksi. Asiakkaan kanssa sovitaan jatkokehityksen lisäksi myös muusta ylläpidosta [Cotterell & Hughes, 2009].

3.2 Scrum

Scrum on 90-luvun alussa kehitetty ketterä projektinhallintamenetelmä. Menetelmä soveltuu niin pieniin kuin isoihinkin projekteihin. Se on kevyt ja helposti omaksuttavissa.

3.2.1 Periaatteet

Keskeisenä periaatteena on päätösten johtaminen kokemusperäisestä tiedosta. Esimerkiksi riskienhallintaa optimoidaan jatkuvasti edellisten kokemusten kautta. Toteutuneet riskit dokumentoidaan myöhempiä projekteja varten. Muita periaatteita ovat läpinäkyvyys, tarkistus ja mukautuvuus [Schwaber & Sutherland, 2013].

Läpinäkyvyydellä viitataan prosessiin ja sen tietosisältöön. Projektin tilan ja sen sisältämien käsitteiden määritysten tulee olla kaikkien projektin sidosryhmien saatavilla. Samoin on tärkeää, että projektissa käytettävien käsitteiden merkitykset ovat kaikille sidosryhmille merkitykseltään samoja [Schwaber & Sutherland, 2013].

Työosioiden valmistuessa projektin tilaa tarkastellaan ja varmistetaan, että tehdyt työosiot tukevat projektin kokonaistarkoitusta. Ohjelmointivirheiden etsimisen lisäksi on tärkeää tarkistaa myös uusien toimintojen tarkoituksenmukaisuus liiketoimintatavoitteiden kannalta. Isoissa ja kriittisissä projekteissa tällaiseen tarkistukseen voidaan määrätä oma työryhmä. Tarkistuksia ei kuitenkaan pidä suorittaa niin usein, että projektin eteneminen hidastuu [Schwaber & Sutherland, 2013].

Mukautumista tapahtuu, kun tarkistuksissa ilmenee epäkohtia. Epäkohdat saattavat liittyä tuotteen laatuun, uusiin vaatimuksiin tai alkuperäisten vaatimusten laiminlyöntiin. Mukautuminen on aina tapauskohtaista ja tarvittavista toimenpiteistä päätetään projektin aikana pidettävissä kokouksissa [Schwaber & Sutherland, 2013].

3.2.2 Roolit

Scrum korostaa projektiryhmän rooleja. Projektiryhmä koostuu tiimeistä, joilla on itsenäiseen työskentelyyn ja päätöksentekoon tarvittavat resurssit. Keskeisiä rooleja tiimissä ovat tuotteen omistaja, kehittäjät ja ScrumMaster.

Omistajan (Product owner) roolissa ovat projektin rahoittajat ja loppukäyttäjät. Omistajan vastuulla on pitää huoli, että kehitettävä tuote soveltuu sen aluperäiseen tarkoitukseen. Projektin aikana vaatimukset ja tarpeet saattavat muuttua ja on tärkeää, että siitä huolimatta projektin päämäärä pysyy samana. Omistaja kirjoittaa käyttötapauksia ja priorisoi ne. Käyttötapauksia käytetään vaatimusmäärittelyn ja kehitystyön pohjana [Gauthier, 2013]. Omistaja hallitsee tuotteen vaatimuksia projektin aikana ja pitää huolen, että vaatimukset ovat yksiselitteisiä ja ymmärrettäviä. Joissain tapauksissa tuotteen omistaja voi delegoida vaatimusten hallinnan kehitystiimille, mutta viime kädessä omistaja on kuitenkin aina vastuussa vaatimusten oikeellisuudesta [Schwaber & Sutherland, 2013].

Kehitystiimi vastaa tuotteen kehittämisestä ja uusien ominaisuuksien liittämisestä lopputuotteeseen jokaisen sprintin yhteydessä. Kehitystiimi koostuu alle kymme-

nestä eri alan ammattilaisesta. Kehitystiimi pystyy itsenäisesti suunnittelemaan, toteuttamaan, testaaman ja dokumentoimaan vaaditut ominaisuudet [Gauthier, 2013].

ScrumMasterin tehtävä on poistaa kaikki hidasteet ja esteet, jotka haittaavat projektin etenemistä. ScrumMaster ei ole tiiminvetäjä, mutta pitää kuitenkin huolen, että kaikki menetelmän sisältämät aktiviteetit tulee hoidettua asianmukaisesti. Samoin päivittäisiin tehtäviin kuuluu myös kehitystiimin motivointi ja pitää huoli, että aktiviteetit palvelevat projektin päämäärää [Gauthier, 2013]. ScrumMaster auttaa tuotteen omistajaa esimerkiksi teknisissä haasteissa, joita saattaa esiintyä vaatimusten hallintatyökalujen käytössä. Lisäksi ScrumMaster varmistaa, että omistajan esittämät vaatimukset pystytään toteuttamaan ja että niiden toteuttaminen on kannattavaa liiketoiminnalle [Schwaber & Sutherland, 2013].

3.2.3 Sprintit

Projektin elinkaari muodostuu sprinteistä. Sprintit ovat muutaman viikon mittaisia ajanjaksoja, jolloin toteutetaan osa vaadituista ominaisuuksista. Kaikki sprintit ovat saman pituisia ja sprintin jälkeen alkaa aina uusi sprintti, kunnes toteutettavia ominaisuuksia ei enää ole [Schwaber & Sutherland, 2013].

Jokaista sprinttiä edeltää tapaaminen, jossa suunnitellaan sprintin sisältö. Suunnitelma sisältää listan toteutettavista ominaisuuksista ja niiden työaika-arvioista. Sprintin suunnitteluun osallistuu koko projektiryhmä. Sprintin pituus voi vaihdella yhden päivän ja yhden kuukauden välillä riippuen projektin koosta. Jokainen sprintti on kuitenkin saman pituinen [Schwaber & Sutherland, 2013]. Luonnollisesti myös suunnittelupalaverin pituus riippuu sprintin pituudesta. Kahden viikon sprintille sopiva palaveri kestää neljä tuntia. Palaverin aikana tarkastellaan tuotteen kehitysjonoa yhdessä tuotteen omistajan kanssa ja kehitystiimi arvioi, miten paljon pystytään toteuttamaan alkavan sprintin aikana [Schwaber & Sutherland, 2013].

Päivittäiset tapaamiset kestävät 15 minuuttia. Tapaamisen aikana jokainen kehitystiimin jäsen kertoo, mitä on saanut aikana edellisen tapaamisen jälkeen ja mitä aikoo tehdä seuraavaan tapaamiseen mennessä. Tässä vaiheessa nostetaan myös esille mahdolliset ongelmat. Tapaamisen tarkoituksena on jakaa tietoa projektin tilanteesta [Schwaber & Sutherland, 2013].

Sprinttiä seuraa aina sprintin arviointitapaaminen. Tapaamisessa käydään epämuodollisesti läpi toteutetut ominaisuudet ja arvioidaan projektin nykyistä tilaa. Samoin käydään läpi vielä toteuttamatta olevat ominaisuudet ja arvioidaan niiden työmäärää verrattuna vielä jäljellä olevaan budjettiin. Tapaamisessa keskustellaan myös sprintin aikana kohdatuista haasteista ja miten haasteet ratkaistiin. Yhtälailla on kuitenkin tärkeää nostaa esille erityiset onnistumiset ja muut positiiviset asiat [Schwaber & Sutherland, 2013].

Tuotteen kehitysjono on listamuotoinen esitys toteutettavan tuotteen ominaisuuksista. Listan ominaisuudet ovat järjestetty. Tietyn ominaisuuden järjestykseen vaikuttavat ominaisuuden antama lisäarvo liiketoiminnalle, riskien määrä ja suuruus sekä prioriteetti. Kehitysjono toimii täydellisenä vaatimusmäärittelynä projektin toteuttajille. Ominaisuuksien määrittely on sitä tarkempi, mitä korkeammalla se on kehitysjonossa. Jonon lopussa olevien ominaisuuksien määrittelyiden ei tarvitse olla yhtä yksityiskohtaisia. Kehitysjonon ylläpidosta huolehtii tuotteen omistaja. Omistaja on vastuussa siitä, että kehitysjonon ominaisuudet tukevat alkuperäisiä vaatimuksia ja asiakkaan liiketoimintatavoitteita. Omistajan tukena kehitysjonon ylläpidossa on kehitystiimi, joka arvioi kaikkien kehitysjonon ominaisuuksien työmäärän [Schwaber & Sutherland, 2013].

3.3 Extreme Programming

3.3.1 Yleistä

Extreme Programming (XP) on projektinhallintamenetelmä, joka määrittelee joukon ohjelmistokehityksen tekniikoita ja prosesseja. Toisaalta XP:tä voidaan käyttää myös pelkkänä viitekehyksenä projektinhallintaan, jolloin aktiviteettien yksityiskohdat määritellään yrityksen tai projektin tarpeiden mukaan. Vaadittuja dokumentteja ja muita projektinhallintaan liittyviä materiaaleja on verrattaen vähän. Samoin rooleja ja aktiviteetteja on suhteellisen vähän. XP:tä soveltavien projektien pituus on vaihdellut yleensä kuuden ja viidentoista kuukauden välillä ja projektiryhmien koko on ollut kahden ja kahdentoista välillä [Dudziak, 2000].

XP rajoittaa projektin läpiviennissä tarvittavien aktiviteettien määrän minimiin. Menetelmän korostamia arvoja ovat yksinkertaisuus, kommunikointi, palaute ja uskallus. Kun aktiviteetit pidetään yksinkertaisina ja työskennellään ryhmässä, on helpompi nähdä projektin hetkittäinen tilanne ja tehdä muutoksia. Ryhmän

keskipisteenä ovat asiakkaan edustajat, jotka työskentelevät päivittäin yhdessä muun ryhmän kanssa [Lowell Lindstrom, 2003].

Myös suunnittelussa tavoitellaan yksinkertaisuutta. Suunnitelma sisältää yksinkertaisesti listan vaadituista ominaisuuksista ja niiden toteutusajankohdista. Toteutusajankohtien takarajat kertovat milloin toteutettujen ominaisuuksien tulee olla testattu ja liitettynä tuotteen tuotantoversioon [Lowell Lindstrom, 2003].

Ohjelmointityö tehdään tavallisesti pareittain. Ohjelmakoodin tulisi olla niin yhdenmukaista, että muutoksia pystyvät tekemään myöhemmin muutkin kuin alkuperäiset toteuttajat. Tärkeä vaihe toteutusta on testaus, jota tehdään jatkuvasti koko projektin ajan [Lowell Lindstrom, 2003].

3.3.2 Projektinhallinta

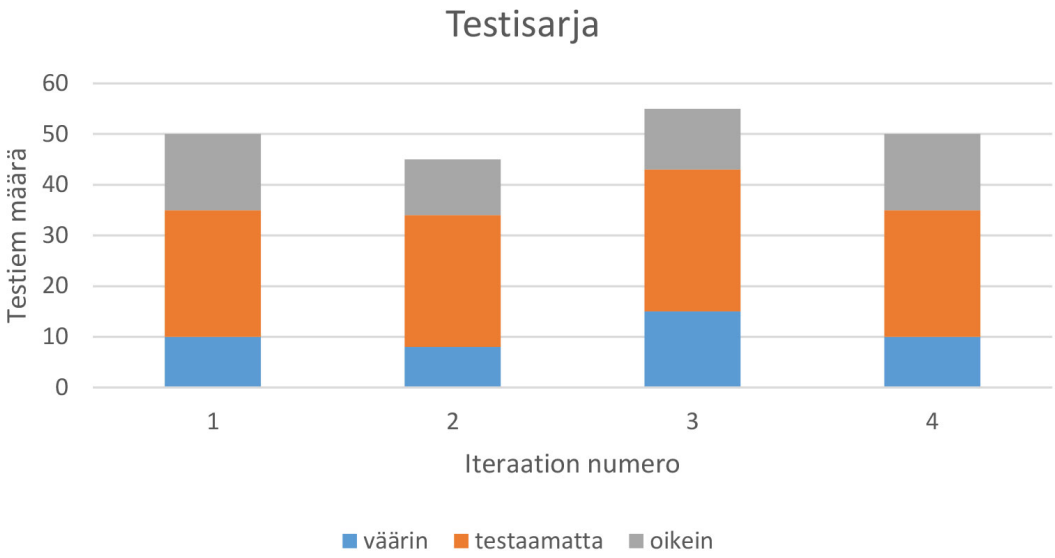
Projektin tilaa arvioidaan kahden muuttujan avulla. Ensimmäinen niistä on kuormitus. Kuormitus mitataan vertaamalla toteutettavien ominaisuuksien työaikaarvioita käytettävissä olevien kalenteripäivien määrään. Hyvällä tiimillä suhde saattaa olla 2,5, joka tarkoittaa, että 8 työtunnin työn tekeminen kestää 2,5 kalenteripäivää [Dudziak, 2000]. Luku saattaa näyttää heikolta, mutta on otettava huomioon, että suuri osa ohjelmointityöstä tehdään pareittain, ja että työpäivään kuuluu ohjelmointityön lisäksi palavereita ja testausta.

Toinen mitattava muuttuja on testisarjan pisteet. Testisarja muodostuu testeistä, jotka asiakas kirjoittaa jokaiselle toteutettavalle ominaisuudelle. Jokaisella ominaisuudella on yksi tai yleensä useampi testitapaus, jolla mitataan ominaisuuden toimivuutta. Kuvassa 5 on ajettu testit neljälle toteutusiteraatiolle, joille asiakas on kirjoittanut testisarjat [Dudziak, 2000].

3.3.3 Roolit

Asiakkaan tulisi hyötyä projektin lopputuotteesta. Asiakkaan tehtäviin kuuluu käyttötapausten ja testisarjojen kirjoittaminen. Asiakas on myös vastuussa vaatimusmäärittelystä [Dudziak, 2000].

Ohjelmoija on XP:n keskeisin rooli. Ohjelmoijan tehtävänä on toteuttaa vaaditut ominaisuudet. Ohjelmoijalta vaaditaan hyviä yhteistyö- ja kommunikointitaitoja,



Kuva 5 Testisarja.

sillä pariohjelmointi on tyypillinen työskentelymuoto XP:ssä. Samoin sosiaaliset taidot ovat tärkeitä, koska XP luottaa tiedon jakamiseen kommunikoinnin kautta pitkien dokumenttien sijaan [Dudziak, 2000].

Valmentajan tehtävä on auttaa toteuttajia tekemään päätöksiä ja tuoda esille vaihtoehtoisia toteutustapoja. Myös valmentajalta odotetaan hyviä yhteistyö- ja kommunikointitaitoja. Valmentajan pitää pystyä selittämään projektin vaiheet ja tilanne asiakkaalle ja johtoryhmälle. Lisäksi valmentajalla tulee olla jatkuvasti kokonaiskäsitys projektin tilasta. Valmentaja voi toimia toisena osapuolena pariohjelmoinnissa [Dudziak, 2000].

Seuraaja on usein sama henkilö kuin valmentaja. Seuraaja seuraa projektin metriikoita, kuten kuormitusta ja testisarjan pisteitä. Mikäli ohjelmoijat eivät erikseen merkitse työtunteja johonkin seuraajan nähtäville, on seuraajan kysyttävä tuntitiedot jokaiselta ohjelmoijalta esimerkiksi muutaman päivän välein. Seuraajan kuuluu huolehtia, että projekti pysyy aikataulussa ja budjetissa [Dudziak, 2000].

Testaaja auttaa asiakasta kirjoittamaan testitapauksia. Yksikkötestit kirjoittavat ja suorittavat ohjelmoijat ohjelmointityön yhteydessä. Yleensä testaajan roolissa onkin sekä ohjelmoijat että valmentaja, eikä testaajan roolissa ole vain yksi henkilö [Dudziak, 2000].

4 Projektinhallintasovellukset

Projektinhallinnassa käytettävien ohjelmien määrä on todella suuri. Käytin tärkeimpänä valintakriteerinä ohjelmille niiden suosiota. Suosion mittarina käytin delicious.com–palvelun kirjanmerkki-viittauksia. Delicious.com on vuonna 2003 perustettu, sosiaaliseen kirjanmerkkaukseen keskittyvä palvelu, jolla on palvelun oman ilmoituksen mukaan yli 6 miljoonaa käyttäjää. Valitut projektinhallintaohjelmat valittiin delicious.com-palveluun tallennetuista kirjanmerkeistä käyttäen hakusanaa project management. Hakutuloksista poimittiin ensimmäiseltä sivulta kaikki projektinhallintaohjelmiin viittaavat kirjanmerkit. Toisena valintakriteerinäni oli se, että ohjelmia oli mahdollista käyttää pilvipalveluna.

Tutkitut ohjelmat olivat hyvin erilaisia, vaikka niiden kaikkien käyttötarkoitus oli sama. Jotkin tutkituista ohjelmista muistuttivat lähinnä monipuolista muistiota tai yksinkertaista tehtävälistaa, kun taas osa ohjelmista oli selkeästi suunnattu yrityskäyttöön ja usean projektin yhtäaikaiseen seurantaan. Toisaalta monipuolisemmat tuotteet olivat poikkeuksetta maksullisia. Vaikka lähestulkoon kaikkiin ohjelmiin pystyi syöttämään useita projekteja, oli niiden seuraaminen ja projektienvälisten päällekkäisyyksien hallinta käytännössä mahdotonta.

4.1 Perustoiminnallisuudet

Projektinhallintasovellusten perustoiminnallisuuksia ovat prosessien ja aktiviteettien hallinta, laadunvalvonta, budjetin ja aikataulun sekä muiden resurssien hallinta.

Tässä kohdassa käsitellään seuraavien sovellusten ominaisuuksia: Action Method [Adobe, 2013], activeCollab [A51 doo, 2013], Redmine [Lang, 2013], Basecamp [37signals, LLC, 2013] ja Pivotal Tracker [Pivotal Labs, 2013].

4.1.1 Projektin tiedot

Pelkistetyimmissä ratkaisuissa (Action Method) projektin tietojen hallinta saattaa tarkoittaa projektin nimeä. Useimmiten lisäksi pystyy määrittämään projektin osakkaat, jotka voivat seurata projektin edistymistä kyseisessä sovelluksessa. Joissain tapauksissa (activeCollab) projektille pystyy perustietojen lisäksi määrittelemään projektipäällikön, asiakkaan, kategorian ja budjetin.

4.1.2 Vaatimusten hallinta

Sisäänrakennetun foorumin tai wikin on tarkoitus toimia ensisijaisena keskustelukanavana. Niiden etuna on, että projektiin liittyvä keskustelu arkistoituu automaattisesti yhteen paikkaan. Käytännössä perinteistä sähköpostia on kuitenkin hankala korvata sen helppouden takia. Foorumin ja wikin käyttö vaatii aina kirjautumisen.

Vaatimusten seurantaa jossain määrin helpottavana ominaisuutena mainittakoon mahdollisuus välittää sähköposteja projektinhallintaohjelmaan [37signals, LLC, 2013]. Ajoittain asiakkaan sähköpostit jäävät vastaanottajansa sähköpostilaatikkoon ja pahimmassa tapauksessa unohtuvat sinne käsittelemättöminä. Tai mahdollisesti sähköpostin sisältämät vaatimukset ja tärkeä tieto jäävät jakamatta muulle projektiryhmälle. Toisaalta sähköpostiviestien välittäminen projektinhallintaohjelmaan kaikkien luettavaksi saattaa lisätä vain ylimääräistä datan määrää ja siten hankaloittaa tiedon hallintaa. Lisäksi sähköposti koetaan usein kahdenkeskiseksi viestintävälineeksi ja viestien jakaminen koko projektiryhmän ja sen osakkaaten kesken saattaa aiheuttaa pahimmassa tapauksessa luottamusongelman.

Toteutustyön aikana vaatimusten tilat muuttuvat. Toteuttajat ylläpitävät heille määrättyjen vaatimusten tiloja projektinhallintasovelluksessa. Tilaa muutetaan sen mukaan, onko työ aloitettu, valmis tai kesken. Tilojen ylläpito on ensisijaisen tärkää projektin seurannan mahdollistamiseksi.

Tutkitut projektinhallintasovellukset toimivat hyvin myös projektin jälkeen esiintyvien ongelmien seurannan apuna. Projektin jälkeen esiintyy usein joitakin ohjelmointivirheitä tai vastaavia ongelmia, jotka edellyttävät toimenpiteitä. Näitä ongelmia voidaan kerätä projektinhallintasovellukseen vaatimusten tavoin. Vastaavasti ongelmien tiloja hallinnoidaan sen mukaan, onko ongelmaa vielä ratkaistu. Ongelmien lisäksi sovellukseen voidaan kerätä uusia vaatimuksia jatkokehitystä varten.

4.1.3 Projektin seuranta

Useimmissa tutkituista projektinhallintasovelluksista on sisäänrakennettu kalenteri. Kalenterin liittäminen sovellukseen on johdonmukaista, sillä tehtäville on mahdollista syöttää valmistumisajankohdan takaraja. Kalenterin avulla tehtävien

aikataulua pystyy seuraamaan. Jos yrityksellä on jo käytössään jokin erillinen kalenterisovellus, saattaa uuden kalenterin käyttöönotto olla turhaa.

Useissa palveluissa ohjelman käyttäjät saavat sähköposti-ilmoituksia projektin tapahtumista (ActiveCollab, Redmine, Basecamp). Monessa palvelussa (ActiveCollab, Basecamp) oli mahdollista seurata projektin tapahtumia RSS-syötteenä. Esimerkiksi uusien työosioiden lisääminen tai vanhojen valmistuminen lisää tapahtuman syötteeseen.

Muutamissa tutkituista ohjelmista (ActiveCollab) pystyi projektia seuraamaan tehtävien lisäksi myös lähdekooditasolla. Lähdekoodia pystyy tarkastelemaan ohjelman käyttöliittymään tuodun versionhallintanäkymän kautta. Käytännössä lähdekoodin integrointi tapahtuu syöttämällä versionhallintasäilön url-osoite, jolloin lähdekoodiin tehdyt muutokset näkyvät myös projektinhallintaohjelmassa. Tämä helpottaa paitsi ohjelmointia vaativien työosioiden seurantaa, myös lähdekoodin laadun arviointia.

Monipuolisemmissa ohjelmissa (ActiveCollab) projektille pystyi määrittelemään tarkistuspisteitä projektin etenemisen seurannalle.

4.1.4 Tehtävienhallinta

Kaikista tutkituista ohjelmista löytyi ainakin jollain tasolla tehtävienhallintamahdollisuus. Ohjelmasta riippuen tehtävälle pystyy syöttämään nimen lisäksi valmistumisajankohdan takarajan ja tehtävästä vastaavan henkilön. Kun tehtävä on suoritettu, merkitsee siitä vastaava sen tehdyksi. Tehtäviä pystyy jakamaan useiden eri henkilöiden kesken ja tehtäviä voi myös merkitä valmiiksi tai poistaa.

Joissain tapauksissa oli vielä mahdollisuus kategorisoida tehtäviä, mikä helpottaa jälkiseurantaa. Joissain ohjelmissa (ActionMethod) ei pystynyt syöttämään tehtävien arvioituja työaikoja, mikä on kuitenkin tuotannon seurannan kannalta välttämätön ominaisuus. Vaatimusten hallintaa tukevana ominaisuutena eräässä ohjelmassa (ActiveCollab) oli mahdollista lisätä tiedostoja työosion kuvaukseen.

Jotkut tutkituista ohjelmista (ActiveCollab, Redmine) toimivat projektinhallintaohjelman lisäksi ongelmienseurannan työkaluna.

4.1.5 Tiedostojen ja resurssien hallinta

Tiedostojenhallintaominaisuudella pyritään organisoimaan kaikki projektiin liittyvä materiaali yhteen paikkaan, josta se on kaikkien saatavilla. Tässä piilee kuitenkin samat ongelmat kuin keskustelussakin. Tiedostot on helpompi lähettää sähköpostilla. Siitä huolimatta tiedostojen lisäysmahdollisuus löytyi kaikista tutkituista ohjelmista. Tiedostojen siirto tehdään ohjelman oman käyttöliittymän avulla eikä erillistä tiedostojensiirto-ohjelmaa tarvita. Merkittävänä ongelmana tiedostojen hallittavuudessa oli usean järjestelmän kohdalla järjestelymahdollisuuden puuttuminen. Jo muutaman kymmenen tiedoston jälkeen dokumenttien eri versioiden hallinta alkaa käydä mahdottomaksi. Vain yhdessä tutkituista palveluista (ActiveCollab) oli mahdollisuus luoda kategorioita tiedostoille ja lisätä tiedostoja työosioiden liitteiksi.

Useimmissa tapauksissa resurssien hallinta tarkoittaa käyttäjätunnusten liittämistä projekteihin ja niiden käyttöoikeuksien määrittelemistä. Käyttäjille voidaan asettaa joko muokkaus- tai pelkät lukuoikeudet. Resurssien, kuten työntekijöiden kuorman seuranta ja hallinnointi on kuitenkin hieman harvinaisempi tehtävä.

Projektinhallintasovelluksissa on mahdollista määrittää käyttäjille eritasoisia oikeuksia. Lukijatason oikeudet mahdollistavat dokumenttien lataamisen, keskustelujen lukemisen ja projektin seurannan. Muokkausoikeudet oikeuttavat tekemään muutoksia projektin työosioihin ja vaatimusmäärittelyyn. Lukijan ja muokkaajan lisäksi voidaan määritellä pääkäyttäjät, jotka pystyvät lisäämään ja poistamaan käyttäjiä.

4.2 Tutkitut sovellukset

4.2.1 Action Method

Action Methodin [Adobe, 2013] käyttö vaatii rekisteröitymisen, joka on ilmainen. Ilmaisen käyttäjätilin voi halutessaan päivittää maksulliseen versioon, jonka hinta on yhdeksän dollaria kuukaudessa. Maksullisen version etuja ovat rajoittamaton tehtävien ja projektien määrä, projektien jakaminen muille käyttäjätileille ja tiedostojen lisäysmahdollisuus.

Projektin tiedot sisältävät vain projektin nimen. Vaatimusmäärittelyä ja muita materiaaleja hallitaan muistioiden, tiedostojen ja keskustelujen avulla. Muistiot

ovat yksinkertaisia projektiin liittyviä tekstielementtejä, joilla ei ole tilaa tai määräaikaa. Tiedostoja voi lisätä niille tarkoitettuun omaan osioon, jossa ne säilyvät järjestyksessä. Tiedostomuotoja ei ole rajoitettu. Keskusteluketjuja voi perustaa useita ja niihin on mahdollista kutsua eri käyttäjiä. Esimerkiksi asiakas voidaan rajoittaa keskustelemaan ainoastaan projektipäällikön kanssa ja samoin projektin ohjelmoijat keskustelevat vain toistensa sekä projektipäällikön kanssa.

Tehtävien tiedot sisältävät työosion kuvauksen, määräajan ja vastuuhenkilön. Kuvaus on lyhyt tekstimuotoinen määrittely tehtävän sisällöstä ja vaatimuksista. Projektin seuranta tapahtuu lokitietojen ja muistutusten avulla. Lokitietoihin tallentuu tiedot valmistuneista tai uusista tehtävistä. Myöhässä olevista tehtävistä ilmestyy muistutuksia käyttäjien sähköposteihin.

Action Method ei varsinaisesti tue Scrumin, XP:n tai PRINCE2:n erityispiirteitä. Sitä voidaan kuitenkin käyttää yleisesti projektinhallinnan apuna projektin tehtäviä ja aikataulua suunniteltaessa [Adobe, 2013].

4.2.2 activeCollab

Samoin kuin Action Methodista, myös activeCollabista [A51 doo, 2013] on tarjolla maksullinen ja maksuton versio. Maksullisen version saa käyttöönsä kertalisenssillä, joka maksaa 499 dollaria. Maksullisella versiolla on lukuisia etuja, joista esimerkkeinä rajoittamaton käyttäjien ja projektin lukumäärä, mobiilikäyttöliittymä, kieliversiot ja lähdekoodin versionhallinnan integrointi. Lisämaksusta saa käyttöönsä vielä käyttötuen ja jatkuvat versiopäivitykset.

ActiveCollabissa projektin tietoja ovat projektin nimi, kuvaus, projektipäällikkö, kategoria, tila, asiakas ja budjetti. Vaatimusmäärittelyä ylläpidetään keskustelupalstan avulla. Keskusteluille on mahdollista määrittää nimi, kuvaus, kategoria ja näkyvyys. Näkyvyyden avulla voidaan rajoittaa keskustelun osallistujia. Esimerkiksi asiakas voidaan rajata tietyistä keskusteluista pois. Keskustelujen lisäksi sovellukseen on mahdollista lisätä tiedostoja. Myös tiedostoja on mahdollista ryhmitellä kategorioihin ja näkyvyydellä voidaan rajata pääsyoikeus vain tietyille käyttäjille.

Projektin tehtävien tietoja ovat tehtävän nimi, kuvaus, kategoria, määräaika, prioriteetti, arvioitu työaika tunteina, tila, liitetiedostot ja vastuuhenkilöt. Lisäksi

tehtäville voi myöhemmin luoda alitehtäviä ja kommentteja. Tehtävään tehtyjä muutoksia voi seurata lokista. Tehtävän tietoihin voi myös kirjata ylös tehtävään käytettyjä työntunteja.

Projektin aikataulua seurataan kalenterin avulla. Kalenterissa näkyy tehtävien valmistumispäivät. Aikataulun lisäksi sovelluksesta löytyy myös resurssien seuranta. Resurssit tarkoittavat käytettyjä työtunteja. Lisäksi projektin etusivulla näkyy loki uusimmista tapahtumista, kuten tehtäviin liittyvistä muutoksista tai uusista keskustelupalstan viesteistä. Ilmoituksista on mahdollista saada myös RSS-syöte.

ActiveCollab tukee monipuolisilla ominaisuuksillaan eri projektinhallintamentelmiä. Sovelluksessa voi valita jokaiselle käyttäjälle roolin lukuisista vaihtoehdoista. Sovelluksen käyttöliittymä ja toiminnot mukautuvat valitun roolin mukaan [A51 doo, 2013].

4.2.3 Redmine

Redmine [Lang, 2013] perustuu vapaaseen lähdekoodiin ja sen käyttö edellyttää sovelluksen lataamista. Sovellusta voi käyttää joko lokaalisti omalta koneelta tai lähdekoodin voi lisätä yrityksen palvelimelle, jossa se toimii pilvipalvelun tapaan. Lähdekoodia voi halutessaan vapaasti kehittää omien tarpeiden mukaan. Vapaan lähdekoodin sovelluksena lisenssimaksuja ei ole. Sovelluksesta löytyy kaikki yleisimmät ominaisuudet.

Projektin tietoja ovat projektin nimi, projektin kuvaus ja kotisivu. Projektiin liittyviä käyttöliittymäkomponentteja on mahdollista rajata projektikohtaisesti. Mahdollisia komponentteja ovat esimerkiksi wiki, keskustelupalsta, Gantt-kaavio ja kalenteri.

Suunnitteluun ja vaatimusmäärittelyyn käytetään projektikohtaisia wiki- ja keskustelupalstoja. Myös wiki- ja keskustelupalstalle voidaan lisätä liitetiedostoja. Projektin vaatimusmäärittelyä helpottaa dokumenttien ryhmittelymahdollisuus. Dokumenteille on sovelluksessa oma osio, johon on mahdollista luoda ryhmiä esimerkiksi sen mukaan onko kyseessä vaatimusmäärittely, palaverimuistio tai projektisuunnitelma.

Projektin tehtävien tietoja ovat tehtävän tyyppi, nimi, kuvaus, tila, prioriteetti, aikataulu, arvioitu työaika ja vastuuhenkilö. Mahdollisia tehtävän tyyppejä ovat bugi, ominaisuus ja taustatehtävä. Tehtäviin on mahdollista liittää tiedostoja.

Seurantaa tukee osaltaan projektikohtainen Gantt-kaavio ja kalenteri. Kalenteriin on mahdollista tehdä omia merkintöjä ja lisäksi kaikkien työosioiden määräpäivät näkyvät automaattisesti. Sovellukseen on myös mahdollista integroida versionhallintajärjestelmä, jonka avulla voidaan seurata ohjelmointityön edistymistä. Aktiviteettinäkymästä nähdään viimeisimmät käyttäjien tekemät toimenpiteet. Toimenpiteitä ovat esimerkiksi viestit foorumeilla, käyttäjien lisäämät tiedostot ja uudet ominaisuudet.

Redmine tukee hyvin esimerkiksi PRINCE2-projektinhallintamenetelmää. Sovelluksen ominaisuuksia voidaan helposti skaalata projektin luontivaiheessa projektin laajuuden mukaan. Projektiryhmän sisältämät käyttäjien roolit on mahdollista syöttää itse. Syötetyille rooleille määritellään käyttöoikeudet. Käyttöoikeuksien rajaus tarkoittaa toimintojen poistoa tietyltä roolilta. Esimerkiksi asiakkaalta poistetaan tyypillisesti kaikki projektin muokkausoikeudet ja muut hallintaan liittyvät toiminnot. Käyttäjää luotaessa valitaan käyttäjälle yksi tai useampi rooli [Lang, 2013].

4.2.4 Basecamp

Basecamp [37signals, LLC, 2013] on maksullinen pilvipalvelu. Basecampin hinnoittelu perustuu projektien määrään ja materiaalien viemään kovalevytilaan. Esimerkiksi 10 projektia ja 3 gb tilaa materiaaleille maksaa noin 15 euroa kuukaudessa. Rajoittamattoman määrä projekteja ja 500 gb tilaa maksaa noin 2350 euroa vuodessa.

Myös Basecamp yrittää siirtää projektiin liittyvän keskustelun ja materiaalin keskitetysti yhteen sijaintiin. Sovelluksesta löytyy kaikki yhteisölliseen työskentelyyn tarvittavat työkalut, kuten kalenteri, keskustelupalsta, tiedostojen jako ja käyttöoikeuksien hallinta. Sovellukseen on mahdollista ladata lisäosia. Lisäosat ovat kolmansien osapuolten toteuttamia, niillä on omat lisenssinsä ja osa niistä on maksullisia. Lisäosia voi myös kehittää itse.

Projektin tiedot koostuvat projektin nimestä ja jäsenistä. Jäsenet jakautuvat kahteen ryhmään, projektin toteuttajiin ja asiakkaan edustajiin. Asiakkaan edustajille näkyvää sisältöä on mahdollista rajoittaa.

Vaatimusmäärittelyä ja suunnittelutyötä varten sovellus tarjoaa keskustelufoorumin. Myös foorumilla tiettyjen viestiketjujen näkyvyyden rajoittaminen asiak-

kaalta on mahdollista. Vaatimusmäärittelyjä voidaan ylläpitää joko erillisissä dokumenttitiedostoissa, jotka lisätään projektiin liitetiedostoiksi tai vaihtoehtoisesti projektiin liittyvissä tehtävälistoissa. Projektin työosioita ylläpidetään tehtävälistoissa. Listoja voi nimetä tehtävien ryhmittelemiseksi. Tehtävän tietoja ovat kuvaus, tila, vastuuhenkilö ja määräaika. Kuvaus on lyhyt, korkeintaan muutaman virkkeen mittainen teksti.

Projektin aikataulua seurataan kalenterista, johon on merkitty kaikkien työosioiden määräajat. Samassa kalenterissa näkee myös kaikkien muiden projektien aikataulut. Kalenteriin voi myös lisätä muita merkintöjä, kuten palavereita. Kalenterin lisäksi sovelluksessa on mahdollista seurata käyttäjien tekemiä toimenpiteitä, kuten uusia viestejä, tiedostoja ja tehtäviä.

Basecamp ei suoraan tue mitään ketteristä projektinhallintamenetelmistä. Sovellus on selkeästi tarkoitettu ennen kaikkea yhteydenpitovälineeksi ja tiedonjakokanavaksi [37signals, LLC, 2013].

4.2.5 Pivotal Tracker

Pivotal Tracker [Pivotal Labs, 2013] on erityisesti ketterään ohjelmistokehitykseen suunnattu pilvipalvelu. Sovelluksen käyttö on maksullista ja hinnoittelu määräytyy käyttäjien määrän, projektien määrän ja materiaalin varaaman kovalevytilan mukaan. Hinnat alkavat 7 dollarista kuukaudessa ja rajoittamattoman projektimäärän viidellekymmenelle käyttäjälle saa 175 dollarilla.

Projektin tietoihin kuuluu vain projektin nimi ja tieto projektin näkyvyydestä. Näkyvyyden voi sallia vain erikseen valituille käyttäjille tai kaikille Pivotal Trackerin käyttäjäille.

Vaatimusmäärittelyä hallitaan sovelluksen kehitysjonon avulla. Kehitysjonoon kirjataan tuotteen ominaisuudet ja muut projektin edellyttämät työtehtävät. Ominaisuuksia voi lisätä kuka tahansa projektin jäsenistä. Ominaisuuden tietoihin tallentuu käyttäjä, joka ominaisuutta on ehdottanut. Lista järjestetään ominaisuuksien prioriteetin mukaan.

Vaatimusmäärittely koostuu käytännössä projektin työosioista. Työosion tietoja ovat kuvaus, tyyppi, laajuus, tila, ehdottaja, omistaja, kommentit, tehtävät ja

hakusanat. Työosion tyyppi voi olla ominaisuus, virhe, tehtävä tai julkaisutoimenpide. Ominaisuudet tuovat asiakkaalle jotain lisäarvoa. Virheet ovat epäkohtia toteutuksessa. Tehtävät ovat välttämättömiä toimenpiteitä, jotka välillisesti edistävät projektia. Julkaisutoimenpiteet ovat palavereita ja muita seurantaan liittyviä tapahtumia. Laajuus kuvastaa työosion haasteellisuutta ja työmäärää. Tila valitaan sen mukaan, onko työosio aloitettu, valmis, toimitettu, hyväksytty tai hylätty. Hyväksynnän tai hylkäämisen tekee asiakas tai asiakkaan edustaja. Ehdottaja on käyttäjä, joka on tehnyt aloitteen työsion lisäämiseksi. Omistaja on työosion toteuttaja. Kommentteja voi lisätä kuka tahansa projektin käyttäjistä. Tehtävät ovat muistilista toimenpiteistä ja alatehtävistä työosion toteuttamiseksi. Hakusanat on tarkoitettu työosioiden ryhmittelemisen helpottamiseksi projektin koon kasvaessa.

Projektia seurataan työosioiden arvioidusta laajuudesta sekä projektin vauhdista. Projektin vauhti lasketaan iteraation aikana toteutettujen työosioiden yhteenlasketun laajuuden avulla. Sovellus laskee vauhdin automaattisesti aikaisempien iteraatioiden perusteella. Vauhdin avulla voidaan ennakoida jäljellä olevien iteraatioiden määrää sekä projektin valmistumisaikataulua.

Ketterät menetelmät ovat huomioitu toteuttamalla kuormituksen seuranta XP:n tyyliin vertailemalla työosioiden työaika-arvioita käytettävissä olevien kalenteripäivien lukumäärään. Muita erityisesti ketteriin menetelmiin suunnattuja työkaluja ovat esimerkiksi Scrum-tyylinen tuotteen kehitysjono ja vaatimusten hallinta tarinoiden avulla [Pivotal Labs, 2013].

5 Projektinhallintasovellusten käytön kehittäminen yrityksessä

5.1 Mediasignal Group

Mediasignal Group on kahdesta pk-yrityksestä muodostuva konserni, johon kuuluvat vuonna 1996 perustettu Mediasignal Communications Oy ja vuonna 2011 perustettu Mediasignal Industrial Oy. Mediasignal Communications Oy tarjoaa internet-ratkaisuja ja viestintäpalveluja pääosin kaupan alan yrityksille sekä julkishallinnon yhteisöille. Mediasignal Industrial Oy:n toimiala on sähköisen liiketoiminnan asiantuntijapalveluiden tarjoaminen teollisuuden ja logistiikan alueilla toimiville yrityksille.

Henkilöstö koostuu seitsemästätoista työntekijästä, joista pääosa on ohjelmoijia. Lisäksi konsernissa työskentelee mm. visuaalisia suunnittelijoita, myyjiä ja sähköisen liiketoiminnan konsultteja. Konsernin päätoimipiste on Tampereella. Lisäksi konsernilla on toimipisteet Helsingissä ja Joensuussa.

Tuotannon toimitila on yhtenäinen esteetön huone, joka avaruudellaan kannustaa yhteistyöhön. Kukin työntekijä saa itse valita työaikansa, kuitenkin niin, että viikottainen työaika ei ylitä neljääkymmentä tuntia, ellei ylitöistä sovita työntekijän ja työnantajan yhteisellä sopimuksella. Yrityksen linjana on, että henkilöstöä ei kuormiteta ylitöillä niin, että ylityöt rasittaisivat työntekijöiden henkistä hyvinvointia. Tähän liittyvä ajatus on, että työntekijät pääsisivät keskittymään heitä motivoiviin oman osaamisalueensa töihin, eivätkä joutuisi käyttämään liikaa aikaansa rutiineihin. Ajatus tukee työhyvinvoinnin lisäksi yrityksen taloudellisten päämäärien saavuttamista. Projektinhallinnan tehostaminen on yksi tämän ajattelun tuottamia käytännön toimenpiteitä.

Vielä vuosituhannen alussa suurin osa yrityksen projekteista oli verkkopalveluiden uudistuksia, jotka tehtiin staattisina html-sivuina. Lopputuloksen ulkoasu ja muu loppukäyttäjälle näkyvä puoli toteutuksesta oli tärkein osa toteutusta. Verkkopalvelun ylläpidettävyys ja muokattavuus ei ollut silloin yhtä tärkeää kuin nykyään ja toisaalta sen koettiin kuuluvan itsestään selvästi palveluntarjoajan erityisosaamiseen ja palvelukokonaisuuteen.

Vuosikymmenen edetessä asiakkaiden tarpeet muuttuivat ja verkkopalveluilta

vaadittiin yhä enemmän ominaisuuksia. Sisällönhallintajärjestelmistä tuli käytännössä välttämättömyys, kun asiakkaat halusivat aikaisempaa enemmän päivittää verkkopalveluidensa sisältöä itse. Loppukäyttäjien web-taidot olivat useimmissa tapauksissa hyvin vaatimattomalla tasolla. Lopulta perinteisten html-sivujen markkinat kehittyivät varsin epäkiitollisiksi kasvuhakuiselle yritykselle – hintakilpailu oli erittäin kovaa ja tekniikka koettiin varsinkin suurempien asiakasyritysten keskuudessa vanhentuneeksi. Mediasignal Communications Oy kehitti muutamia vuosia omaa sisällönhallintajärjestelmäänsä, mutta asiakaskunnan vaatimusten yhä laajentuessa yhtiö siirtyi avoimen lähdekoodin järjestelmiin. Samalla myynnin painopiste siirtyi lisenssien myynnistä asiantuntijapalvelujen myyntiin. Laaja julkaisujärjestelmä sisältyi nyt toimituksiin miltei vakioelementtinä.

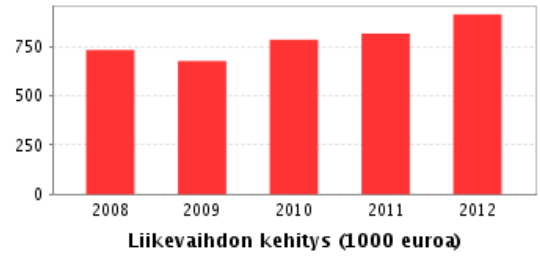
5.2 Yrityksen toiminnan mittarit

Mediasignal Communications Oy:ssä yrityksen johtohenkilöt seuraavat liiketoiminnan kehitystä päivittäin erilaisten mittarien avulla. Mittareiden seuraaminen on tärkeää, jotta ongelmiin pystytään reagoimaan ajoissa. Yrityksessä on pyritty suosimaan kvantitatiivisia mittareita kvalitatiivisten sijaan. Kvantitatiiviset mittarit ovat tarkempia. Oleellista ei ole esimerkiksi mitata, onko liikevaihto kasvanut vai laskenut. Sen sijaan tärkeää on mitata tarkalla tasolla, mistä palvelujen ja tuotteiden osa-alueista liikevaihto ja sen kehitys milloinkin muodostuu.

Liikevaihtoa ja tulosta mittaamalla voidaan tutkia yrityksen taloudellista kehitystä. Taloudellista kehittymistä voidaan arvioida suhteuttamalla omat tulokset kilpailijoiden vastaaviin. Liikevaihto muodostuu yrityksen myyntituotoista. Kuvassa 6 näkyy liikevaihdon kehitys Mediasignal Communications Oy:ssä. Muita tärkeitä taloudellisen tilan mittareita ovat liikevaihdon muutos, tilikauden tulos ja liikevoitto.

5.2.1 Myynnin seuranta

Mediasignal Groupissa on neljä myyjää, jotka hankkivat uusia asiakkuuksia. Lisäksi konserni kehittää jälleenmyyntiverkostoa. Tämän tutkielman kirjoitusvaiheessa kehitystyö oli alkuvaiheessa ja yksi jälleenmyyntisopimus oli solmittu. Myyjien toiminnassa uusien asiakkaiden hankinta tapahtuu yhteydenotolla puhelimitse tai sähköpostitse. Mikäli asiakas on kiinnostunut yrityksen palveluista, sovitaan



Kuva 6 Mediasignal Communications Oy:n liikevaihdon kehitys. Lähde Suomen Asiakastieto.

mahdollisesti palaveri, jossa asiakas kertoo tarkemmin tarpeistaan. Myynnin tavoitteena on löytää ratkaisumahdollisuuksia asiakkaan tarpeisiin ja toimittaa asiakkaalle pääsääntöisesti työaika-arvioon perustuva tarjous.

Toinen mahdollinen tarjouksen lähetykseen johtava heräte on tarjouspyyntö, joita asiakkaat voivat tehdä puhelimitse, sähköpostilla tai suoraan yrityksen verkkosivuilta. Asiakkaan kanssa yritetään aina sopia palaveri. Palavereissa asiakkaan tarpeet kyetään kartoittamaan paremmin, ja sen jälkeen tarjous voidaan laatia vastaamaan asiakkaan todellisia tarpeita. Asiakkaan tarjouspyyntö ei aina huomioi kaikkia mahdollisuuksia tai painottuu detaljeihin kokonaisuusratkaisutarpeen jäädessä hämärän peittoon. Palaverin avulla vähennetään väärinkäsitysten riskejä.

Suurin osa yrityksen liikevaihdosta muodostuu hyväksyttyjä tarjouksia seuraavista projekteista. Muita tulonlähteitä ovat palvelutuotteiden kuukausimaksut sekä vanhojen verkkopalveluiden päivitys- ja muutostyöt. Niiden osa on kuitenkin hyvin pieni verrattuna uusien projektien tuottoon. Siten tarjousten määrä ja niiden tuomien tilausten määrä ovat ehdottomasti yksi tärkeimpiä Mediasignal Communications Oy:n mittareista. Tehtyjen tarjousten kokonaismäärästä pidetään kirjaa. Tarjouksia on eroteltu vielä tarkemmin usealla eri tavalla:

- Tarjoukset uusille asiakkaille.
- Tarjoukset vanhoille asiakkaille.
- Myynnin aloitteesta tehdyt tarjoukset.
- Tarjouspyyntöjen kautta tehdyt tarjoukset.
- Tapaamiseen johtaneet tarjoukset.

5.2.2 Asiakkaan tyytyväisyydestä huolehtiminen

Asiakastyytyväisyydessä on kyse nykyisten asiakassuhteiden huolehtimisesta. Hyvin onnistunut projekti kannustaa asiakasta jatkamaan yhteistyötä saman toimittajan kanssa. Asiakkaan tyytyväisyyteen vaikuttavat projektin vastuuhenkilöiden onnistuminen asiakaspalvelutyössä, erityisesti yhteydenpidossa, sekä projektikokonaisuuden onnistuminen erityispiirteinään projektin mahdolliset aikataulunmuutokset, budjetinmuutokset, tuotteeseen liittyvien vaatimusten täyttäminen, tuotteenlaatu ja asiakkaan ensimmäiset käyttökokemukset.

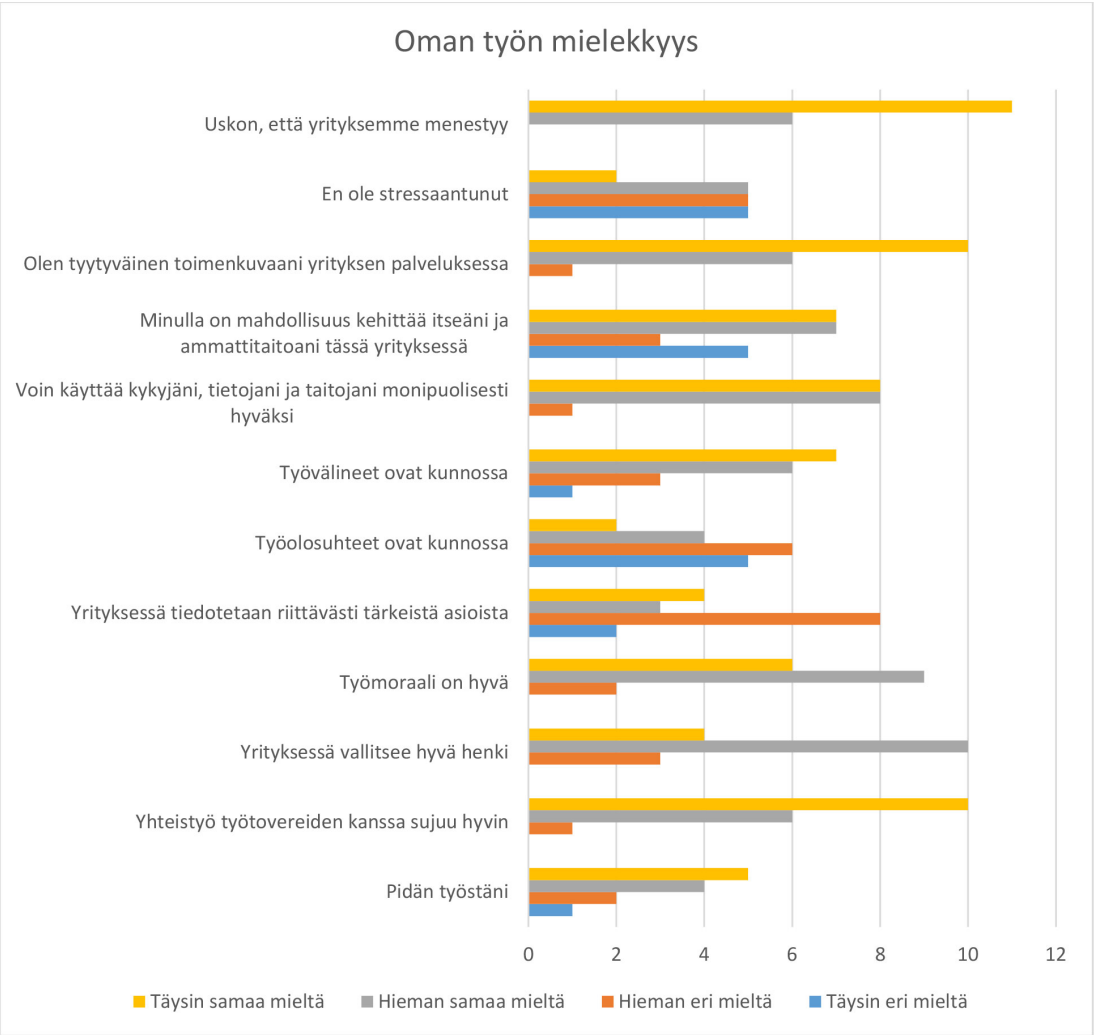
Asiakastyytyväisyyttä mitataan säännöllisillä asiakastyytyväisyyskyselyillä. Pitkäaikaisten asiakkaiden kanssa järjestetään vuosittain katselmuksia, joissa arvioidaan vuoden aikana valmistuneiden projektien onnistumista sekä tulevia projekti- ja resursointitarpeita. Jokaisen projektin jälkeen järjestettävässä päätöspalaverissa käydään läpi toteutetun tuotteen toiminnallisuus ja tarkistetaan ominaisuudet alkuperäisiä vaatimuksia vasten. Tämän lisäksi uusille asiakkaille lähetetään kyselylomake, jonka kysymykset liittyvät projektin läpivientiin. Mikäli asiakas toivoo myös tämän kyselyn tulosten yhteistä läpikäpikäyntiä, sellainen järjestetään.

5.2.3 Henkilöstön hyvinvointi ja tuotannon tehokkuus

Henkilöstön tyytyväisyyttä mitataan puolen vuoden välein käydyissä kehityskeskusteluissa. Tyytyväisyys ja viihtyvyys ovat yksi tarkimmin seuratuista asioista Mediasignal Communications Oy:ssä. Henkilöstön hyvinvoinnilla on yrityksessä todettu olevan välitön vaikutus kaikkiin muihin yrityksen menestykseen vaikuttaviin tekijöihin. Tyytyväisyyttä mittaavat kysymykset ovat monivalintakysymyksiä, joten niistä on mahdollista muodostaa kuva 7. Kehityskeskustelun lisäksi tyytyväisyyttä voidaan mitata henkilöstön vaihtuvuudella ja poissaolojen määrällä.

5.2.4 Tuotannon tehokkuus

Mediasignal Communications Oy:ssä ajatellaan, että tuotannon tehokkuus ja projektinhallinnan onnistuminen ovat suoraan yhteydessä henkilöstön tyytyväisyyteen. Tuotannon tehokkuutta voidaan arvioida mittaamalla tuotannon läpäisykykyä. Läpäisykyvyllä tarkoitetaan samanaikaisten projektien määrää. Läpäisykyky vaihtelee projektien koon ja monimutkaisuuden mukaan. Jonain aikana tuotannossa



Kuva 7 Henkilöstön tyytyväisyys Mediasignal Communications Oy:ssä.

saattaa olla iso projekti, joka sitoo kaikki ohjelmoijat samanaikaisesti. Toisaalta saattaa olla tilanne, että jokaisella työntekijällä on oma projektinsa menossa. Toinen tapa arvioida tuotannon tehokkuutta on mitata projektien läpimenoaikoja. Projektin läpimenoaika mitataan jakamalla projektin käytettävissä olevat kalenteripäivät budjetoitujen tuntien määrällä.

Mediasignal Communications Oy:ssä projektien koko vaihtelee muutamasta tunnista yli 600 tuntiin. Keskimääräinen projektin työmäärä on noin 30 tuntia [Mediasignal Communications Oy, 2013].

Useimmat yrityksen projektit ovat kestoltaan noin kuukauden mittaisia [Media-

signal Communications Oy, 2013]. Lyhyen elinkaaren vuoksi pienetkin viivästykset aiheuttavat suhteellisen ison muutoksen aikatauluun. Mediasignal Communications Oy:ssä aikataulujen pitävyys on tärkeä ehto projektin onnistumiselle paitsi asiakastyytyväisyyden näkökulmasta, myös yrityksen projektien ja talouden hallinnan kannalta.

5.2.5 Ohjelmiston laadun tarkkailu

Projektin kustannusten muuttuminen kesken projektien on tyypillinen ongelma kaikille keskimääräistä suuremmille projekteille. Nämä projektit pääosin laajenevat kesken toteutuksen. Laajeneminen johtuu useimmiten asiakkaan ilmoittamista lisätarpeista. Keskimääräistä kustannusten muutosta voidaan seurata Mediasignal Communications Oy:ssä vertaamalla projektien alkuperäisiä arvioita lopulta laskutettuihin summiin. Keskimääräinen kustannusten muutos on otettava huomioon myös tuotannon läpäisykykyä arvioitaessa. Kustannusten muutosten seurannassa oleellisinta olisi löytää se perimmäinen syy, josta muutokset aiheutuvat.

5.3 Projektien läpivienti

5.3.1 Projektinhallintamalli

Yrityksen vanha projektinhallintamalli laadittiin yrityksen elinkaaren alkuvaiheessa, kun suurin osa liikevaihdosta tuli vielä staattisten html-sivujen toteutuksista. Nykyisen projektinhallintamallin perusta, työn jakaminen tekijöiden kesken, on ollut yrityksen käytössä vuodesta 1996 lähtien, mutta hallinnan vaatimukset ovat monipuolistuneet erittäin paljon. Kehitykseen on vaikuttanut myös asiakkaan projektikriteerit, varsinkin yhtiön asiakaskunnan kehittyessä kattamaan myös suuryrityksiä.

Vanha projektinhallintamalli sisältää neljä tai viisi vaihetta riippuen projektista. Ensimmäinen vaihe on aina vapaamuotoinen palaveri, jossa kartoitetaan asiakkaan tarpeet ja projektin tavoitteet. Palaveriin osallistuu asiakkaan edustajat, usein graafinen suunnittelija ja mahdollisesti tekninen asiantuntija. Keskustellut asiat kirjataan palaverimuistioon, jota graafinen suunnittelija käyttää suunnittelutyön apuna. Tässä vaiheessa tarkennetaan myös tarjousvaiheessa ensimmäisen kerran arvioitua aikataulua, joka muodostetaan yleensä käänteisessä järjestyksessä julkaisupäivästä taaksepäin.

Toinen vaihe sisältää verkkopalvelun ulkoasun ja käyttöliittymän suunnittelun kuvankäsittelyohjelmalla. Vaihe sisältää graafikon ja teknisten toteuttajien välistä kommunikointia. Valmis malli lähetetään asiakkaalle kommentoitavaksi. Malli on vielä tässä vaiheessa staattinen ja toiminnalliset ominaisuudet on kuvattu tekstinä. Asiakkaan kommenttien ja niitä seuraavan yhteydenpidon perusteella tehdään mahdolliset muutokset suunnitelmiin. Kun muutokset on tehty ja asiakas on hyväksynyt suunnitelmat, voidaan projekti siirtää jatkotuotantoon.

Kolmannessa vaiheessa suunnitelmien pohjalta tehdään html-tiedostot. Tämä vaihe on hyvin suoraviivainen ja vaatii harvoin asiakkaan panosta. Käytännössä asiakkaan ohjeistus ei juuri koskaan käsittele tätä vaihetta, ja ohjelmoijat laativat sivuston teknisen toteutuksen, joka on valittu yrityksen projektikonseptiin pitkän kokemuksen pohjalta. Ohjelmoijilla on myös vapaus muuntaa työtä omaan työtapaansa soveltuvaksi, mutta lopputuloksen on täytettävä konseptiratkaisun peruskriteerit. Asiakkaan roolina tässä vaiheessa on toimittaa verkkopalvelun sisällöt, kuten kuvat ja teksit.

Neljännen vaiheen sisältö riippuu projektista. Mikäli verkkopalvelu sisältää teknisiä osia, kuten lomake- tai flash-elementtejä, toteutetaan ne tässä vaiheessa. Nykyisissä projekteissa nämä elementit ovat itsestäänselvyys. Projektin viimeinen vaihe ennen hyväksyttäväksi lähettämistä on testaus. Useimmiten projekteissa testataan syntynyttä lopputulosta teknisen toteutuksen ja viestinnän osalta. Seuraavaksi toteutus lähetetään hyväksyttäväksi asiakkaalle. Kun asiakas on antanut hyväksyntänsä, voidaan verkkopalvelu julkaista ja projekti on valmis.

5.3.2 Käytetyt projektinhallintasovellukset

Aivan Mediasignal Communications Oy:n toiminnan alkuvaiheissa keskeisenä projektinhallintasovelluksena toimi Word-tekstinkäsittelyohjelma, johon tekijät kirjasivat laskutettavat työtuntinsa tekstimuodossa. Osin tunteja ilmoitettiin laskuttajalle sähköpostilla. Laskutus tapahtui ohjelmalla, johon tietoja ei voinut siirtää kopioimalla, joten laskuttaja kirjoitti laskutustiedot uudelleen laskulle. Laskut tulostettiin, pakattiin kirjekuoriin ja lähetettiin asiakkaalle postitse. Seuraavassa vaiheessa siirryttiin käyttämään Excel-taulukkoa. Tekijät merkitsivät työnsä omiin tiedostoihinsa. Sen jälkeen tehdyt työt koottiin keskitetysti yhteen tiedostoon. Kuukauden päätteeksi tiedosto toimitettiin laskuttajalle, joka edelleen kirjoitti erikseen laskun sisältötiedon laskutusohjelmaan. Laskutus tapahtui yhä manuaalisesti.

Seuraavassa vaiheessa vuonna 2002 yritys otti käyttöön sähköisen laskutuksen ja uuden laskutusohjelman. Nyt eri asiakkaiden laskutustiedot voitiin siirtää laskutusohjelmasta kertasiirtona pankin sähköisen laskutuksen palveluun, joka huolehti laskujen lähettämisestä asiakkaille. Haasteena oli yhä työtietojen hajanainen merkintäprosessi. Tietoja ei enää pitänyt erikseen kirjoittaa laskutusohjelmaan, mutta kunkin asiakkaan laskutettavat työtiedot piti kopioida asiakas kerrallaan manuaalisesti Excelistä laskutusohjelmaan. Yhtiö tutki tässä vaiheessa eri projektinhallintasovelluksia menettelyn automatisoinniksi ja selkeyttämiseksi mahdollisimman pitkälle. Vuonna 2007 yhtiö päätyi hankkimaan lisenssit kaupalliseen projektinhallintasovellukseen. Vajaan vuoden käytön jälkeen todettiin, että ohjelmisto oli liian keskeneräinen ja kankea yhtiön käyttötarkoituksiin ja että toimittajan lupaamia päivityksiä ei koskaan saatu. Ohjelmiston käytöstä jouduttiin luopumaan ja palaamaan vanhaan Excel-käytäntöön. Tässä vaiheessa itse töiden merkintäprosessia selkeytettiin ja ratkaisulla kyettiin toimimaan toistaiseksi. Jo tuolloin oli kuitenkin selvää, että menettely on yhtiön kasvaessa ongelmallinen ja ratkaisua on kehitettävä.

5.4 Koetut ongelmat yrityksessä

5.4.1 Projektinhallintasovellukseen liittyvät ongelmat

Laskutustiedostoa pystyi käsittelemään vain yksi henkilö kerrallaan. Tämä muodostui erittäin suureksi haasteeksi henkilöstömäärän kasvaessa. Lisäksi laskutukseen siirtyvien työmerkintöjen yhtenäistäminen valmiiden kirjauspohjien avulla osoittautui käytännössä vaikeaksi. Myös vanhojen työtietojen tarkistaminen oli kankeaa esimerkiksi silloin, jos laskutusvirheen takia asiakkaalle oli lähetettävä hyvityslasku ja selvitys virheestä.

5.4.2 Projektinhallinnan ongelmat

Henkilöstömäärän sekä projektien koon ja haastavuuden kasvaessa myös hallinnan ja seurannan tarve on kasvanut. Excel-käytännössä monipuolinen seuranta oli mahdotonta. Lisäksi avustajat merkitsivät tuntityötietonsa kahteen paikkaan. He merkitsivät ne ensin Exceliin tai lähettivät tuotantopäällikölle. Kuukauden päätteeksi he toimittivat tauloushallintoon koosteen tekemistään työtunneista palkkion maksua varten. Joskus tiedoissa oli eroja ja niiden vertaaminen oli työlästä.

Taloutta seurattiin lähinnä laskutusohjelman avulla, mikä tarkoitti historian seuraamista. Esimerkiksi tarjousmäärien ja niiden tyypin seuranta oli erittäin vaikeaa, sillä koostettua tietoa ei ollut saatavissa automaattisesti.

Myöskään tilattujen, tuotantoon tulevien projektien määrästä ei ollut yhtä, päivittyvää tietoa. Sekä myyntiin että tuotantoon liittyvä yksittäinen vaikeus oli se, että myyjillä ei ollut tarkkaa tietoa olemassa olevien projektin sovituista aikatauluista, kun he sopivat asiakkaiden kanssa uusien projektien aikatauluista.

Tämä korostui siinä, että olemassa olevien projektien aikataulut muuttuivat melko usein ja tieto muutoksista oli tuotannossa, eikä se ei tullut useinkaan myyjille saakka. Näin päällekkäisyyksien ja tuotantoruuhkien mahdollisuus oli suuri ja myös realisoitui usein.

5.5 Projektinhallintajärjestelmän käyttäjät ja vaatimukset

Projektinhallintaohjelman toteutuksen mielenkiintoisista haasteista antaa kuvan sen osakkaiden ja heidän tarpeidensa kirjavuus.

Tuotantopäällikön vaatimukset pysyvät suurelta osin samana kuin ennenkin. Työntekijöiden kuormitusta pitäisi kuitenkin pystyä seuraamaan entistä tarkemmin ja pidemmälle tulevaisuuteen. Myös hallituksen jäsenillä täytyy olla järjestelmässä oma osio, josta he pystyvät seuraamaan yrityksen tulostavoitteiden mittareita. Toimitusjohtajalla on käytössään samat työkalut kuin hallituksella ja tuotantopäälliköllä. Toimitusjohtajan täytyy saada tietoa talousseurannan ohella mm. henkilöstöresurssien käytöstä ja tuotannon tehokkuudesta.

Projektien toteuttajilla (ohjelmoijat ja graafikot) täytyy olla mahdollisimman selkeä tilannekuva projekteista. Käytännössä heidän pitää tietää vähintään kuluvalla viikolla toteutettavat työosiot sekä niiden arvioidut työmäärät. Heillä on myös oltava käytettävissään kaikki saatavilla oleva tieto työosioiden yksityiskohdista. Heidän täytyy voida merkitä työpäivän aikana käytetyt tunnit nopeasti ja vaivattomasti tuotantotilanteiden yleistä seurantaa sekä laskutusta varten. Tavoitteena on, ettei heillä kuluisi uuden projektinhallintaohjelman käyttöön sen enempää aikaa kuin ennenkään. Toiveena on merkintöihin kuluvan ajan väheneminen entisestään.

Myyjien tarpeet projektinhallintajärjestelmässä koskevat sen asiakkuudenhallintaosiota. Asiakkuudenhallinnan lisäksi heillä täytyy olla saatavilla tieto nykyisille

asiakkaille aiemmin tehdyistä projekteista ja mahdollisista aiemmin tehdyistä tarjouksista. Samoin heillä täytyy olla saatavilla tieto aikaisemmin toteutettujen projektien ja työosioiden työmääristä. Näin tarjousten tekeminen helpottuu kun nähdään paljonko vastaavantyyppiseen työhön on aikaisemmin kulunut aikaa ja mistä elementeistä ajankulu on koostunut. Myyjien pitää myös pystyä välttämään päällekkäisyyksiä asiakaskontaktoinnissaan ja tarjouksissaan näkemällä toisien myyjien hallinnoimat asiakkuudet.

Taloushallinnan tehtävä on huolehtia töiden laskuttamisesta. Laskutettaviin töihin kuuluvat projektit, pienet päivitystyöt ja erilaiset kuukausimaksut. Suuret projektit laskutetaan osissa ja niiden budjetti saattaa kasvaa kesken toteutuksen, tämä on tavanomaista. Pienet päivitystyöt laskutetaan muutaman kerran kuussa ja mikäli samalla asiakkaalla on kertynyt useita töitä, on ne saatava samalle laskulle. Kuukausimaksut kertyvät erilaisista ylläpitopalveluista, kuten palvelimien, domainien ja ennen kaikkea toteutettujen verkkopalveluiden ylläpidosta. Kuukausimaksujen laskutusvälit vaihtelevat asiakkaiden ja palvelukuvauksen mukaan yhdestä kuukaudesta kahteen kuukauteen.

Projektinhallintajärjestelmän tulee mukautua Mediasignal Communications Oy:n kasvuun ja muutoksiin. Niinpä uuden projektinhallintajärjestelmän ylläpidon tulee olla mahdollisimman helppoa. Lisäksi on tärkeää, että muutoksia pystyy tekemää kuka tahansa yrityksen ohjelmoijista. Ylläpitäjälle onkin ensisijaisen tärkeää, että järjestelmän toteutuksessa on käytetty mahdollisimman pitkälti standardin mukaisia ratkaisuja, ja että lähdekoodi ja sen konventiot on tarkasti dokumentoitu. Aikaisemman järjestelmän yhdeksi ongelmaksi muodostui samojen komponenttien useat käyttötarkoitukset. Esimerkiksi asiakkaiden hallitseminen kävi kaikille hankalaksi, kun samaa käyttöliittymää käyttivät toteuttajat, myyjät ja toimitusjohtaja, kaikki eri tarkoituksiin. Lopulta käyttöliittymä ei palvellut kenenkään tavoitteita tyydyttävällä tasolla. Uudessa projektinhallintajärjestelmässä haluttiin toteuttaa kaikkien edellä mainittujen käyttäjätyyppien tarvitsema toiminnallisuus erillisinä moduuleina. Näin ylläpito helpottuu huomattavasti ja esimerkiksi tuotantopäälliköiden asiakasnäkymä ei täyty vain myyjien tarvitsemista toiminnallisuuksista.

5.6 Valmiit vaihtoehdot

Vaatimusten pohjalta testattiin kahta avoimen lähdekoodin vaihtoehtoa ja tutustuttiin yhteen kaupalliseen järjestelmään. Avoimen lähdekoodin vaihtoehdot to-

dettiin kuitenkin hyvin pian täysin riittämättömiksi Mediasignal Communications Oy:n tarpeisiin. Kaupallisen tuotteen kohdalla esteeksi tuli tuotteen keskeneräisyys ja kauppiaan lupaamien päivitysten viivästyminen, joten siitäkin luovuttiin.

Suurin yhteinen ongelma valmiiden ratkaisujen välillä oli seurannan puute. Yksittäisten projektien seuranta oli joissain tapauksissa mahdollista, mutta kaikista puuttui kaikki käynnissä olevat projektit yhdistävä näkymä, josta tuotannon tilaa voisi seurata. Tuotannon kannalta on erittäin tärkeä hallita projektien ja niiden työosioiden limittymistä tietyllä aikavälillä, esimerkiksi kuukausittain. GANTT-kaavio on eräs suosittu työkalu projekin seuraamiseen, mutta siitäkin näkee vain yhden projektin kerrallaan.

Lopulta luovuttiin valmiiden ratkaisujen etsimisestä ja päädyttiin toteuttamaan oma projektinhallintajärjestelmä alusta asti itse.

5.7 Valmiin pilvipalvelun pilottiprojektit

Mediasignal Communications Oyssä otettiin Basecamp koekäyttöön ja arvioitiin sen soveltuvuutta yrityksen tarpeisiin. Tavoitteena oli lähentää asiakkaan ja muun projektiryhmän yhteydenpitoa ja avata projektin materiaalit ja keskustelut kaikkien sidosryhmien ulottuville. Basecampin käyttöönotto oli helppoa, sillä pilvipalveluun tarvitsi vain rekisteröityä ja maksaa ensimmäinen kuukausimaksu, jonka jälkeen käytön voi aloittaa.

Käyttöönoton jälkeen seuraavat isot projektit syötettiin Basecampiin hallittaviksi. Käyttöliittymä oli niin yksinkertainen, että vähemmän teknisetkin asiakkaan edustajat pystyivät käyttämään ohjelmaa tehokkaasti. Jokaisessa tapauksessa asiakkaalle pidettiin kuitenkin lyhyt opastus puhelimitse ohjelman käytöstä ja tunnusten luonnista. Projektipäällikön ja asiakkaan edustajien lisäksi Basecamp-projektiin liitettiin sovelluskehittäjät, graafikot ja johtoryhmä.

Yksi Basecamp-projekteista oli tarkoitettu vain yrityksen sisäiseen käyttöön. Tarkoituksena oli hallita projektin tehtäviä. Tehtävien tiedot syötettiin Basecampiin ja niiden tiloja ja työaikoja hallittiin siellä hetken aikaa projektin alkuvaiheessa. Ylläpitoa alettiin kuitenkin laiminlyömään ja lopetettiin kokonaan projektin edetessä. Käyttäjien haastattelussa kävi myöhemmin ilmi, että ohjelmoijat ja graafikot eivät kokeneet tehtävien ylläpitoa tarpeellisena. Jokaisella oli kuitenkin

omat tehtävälistansa paperilla tai sähköisenä omavalintaisessa muodossa, joiden käyttö oli helpompaa ja nopeampaa.

Toisessa Basecamp-projektissa asiakkaallakin oli käyttäjätunnus ohjelmaan. Tavoitteena oli siirtää keskustelu ja materiaali keskitetysti yhteen paikkaan. Myös tässä tapauksessa ohjelman käyttö väheni asteittain. Viestiketjujen ja tiedostojen määrän kasvaessa turvauduttiin yhä useammin perinteisiin keinoihin, kuten puhelimeen tai sähköpostiin. Ennen ohjelman käytön lopettamista, asiakas lähetti materiaaleja sähköpostilla projektipäällikölle, joka lisäsi ne ohjelmaan asiakkaan puolesta.

Johtopäätöksenä Basecamp-ohjelman käytöstä Mediasignal Communications Oy:ssä, voidaan todeta käytön olevan hankalaa, kun materiaalia on paljon. Yhdessä tapauksessa ohjelman käytöstä jouduttiin luopumaan heti käyttöönoton yhteydessä, kun ilmeni, että asiakkaalla oli liian vanha selainversio. Asiakasyrityksen suuresta koosta ja it-osaston politiikasta ja hallinnollisista asioista johtuen selainversion päivitys ei ollut mahdollista.

5.8 Päätös projektinhallintaohjelmiston rakentamisesta itse ja toteutukseen valitut ominaisuudet

Yhtiössä, joka oli vuonna 2011 perustetun uuden osakkuusyrityksen myötä muuttunut tässä vaiheessa konsernimuotoiseksi toimijaksi, luovuttiin valmiiden ratkaisujen etsimisestä. Projektinhallintajärjestelmä päädyttin toteuttamaan alusta asti itse. Tämän tutkielman tekijä valittiin projektin vastaavaksi ohjelmoijaksi. Järjestelmän toteuttamisessa sisäisenä projektina on lukuisia etuja. Tärkeimpinä päätökseen vaikuttavana tekijänä oli järjestelmän muunneltavuus. Itse tehdyssä järjestelmässä ei ole yhtään tarpeetonta ominaisuutta, mutta kuitenkin kaikki tarvittava ja riittävällä tarkkuudella. Lisäksi järjestelmän on mahdollista mukautua yrityksen liiketoiminnan kehitykseen ja kasvaa yrityksen mukana.

Toisaalta oman järjestelmän kehittämisessä on varjopuolensakin. Kehittäminen vie aikaa ja joidenkin sellaisten ominaisuuksien, jotka löytyvät kaikista valmiista ratkaisuista, toteuttaminen itse saattaa olla todella työlästä ja aikaa vievää, esimerkiksi käyttäjien- ja käyttöoikeuksienhallinta.

5.8.1 Projektin ja työosioiden tiedot

Projektin tietoihin on lisätty kaikki oleellinen tieto, jota tarvitaan projektin seurantaan ja laskutukseen. Perustietojen, kuten projektin nimen, kuvauksen, asiakkaan, tilan, projektipäällikön budjetin ja aikataulun lisäksi projekti on mahdollista luokitella päivitystyöksi. Päivitystyö eroaa projektista siten, että se laskutetaan heti eikä sillä ole työosioita. Päivitystöitä tulee luonnollisesti aina projektien päättymisen jälkeen ja niiden laskutus tehdään yksittäin.

Lisäksi laskutuksen kannalta oleellisina tietoina projektille tulee antaa tuntihinta ja laskutussuunnitelma. Laskutussuunnitelman muodostaa joukko rivejä, joilla on laskutettava summa sekä päivämäärä. Laskutussuunnitelma mahdollistaa projektin laskuttamisen osissa. Nämä ovat kaikki taloushallinnon ja liiketoiminnan kannalta tärkeitä ominaisuuksia.

Projektin sisältämillä tehtävillä eli työosioilla on paljon samankaltaisia piirteitä kuin projektillakin. Työosioillakin on vastuuhenkilöt, aikataulu, tila ja budjetti.

5.8.2 Tilastot ja seuranta

Mediasignal Communications Oy:n hallitus edellyttää tarkkoja seurantamahdollisuuksia liiketoiminnan mittareille. Päivittäin käytössä oleva seurannan työkalu on kuukausinäkymä, joka on listanäkymä tuotannossa olevista projekteista ja niiden budjeteista. Näkymän avulla seurataan yrityksen taloudellista tilannetta ja ennustetta.

Projektien lisäksi samassa näkymässä on yrityksen asiakkaiden kanssa tekemät kuukausilaskusopimukset. Kuukausilaskut koskevat yleensä jonkin yrityksen tuotteen lisenssiä tai palvelinmaksua. Nykyisen ja tulevan taloudellisen hyvinvoinnin seurannan lisäksi myös laskutushistorian seuranta on mahdollista. Menneitä projekteja ja päivitystöitä voi hakea niiden arkistosta. Asiakaskohtaisesti voi selata tehtyjä töitä. Tehtyjen töiden seuranta helpottaa tulevien töiden työaikojen arviointia. Tehdyistä töistä voi verrata arvioitua ja toteutunutta budjettia keskenään.

5.8.3 Laskutus

Projektinhallinnan ja seurannan työkalujen lisäksi uudesta projektinhallintajärjestelmästä löytyy paljon toiminnanohjausjärjestelmään liittyviä elementtejä. Lasku-

jen hallinnointi ja lähetys hoidetaan ulkoisen järjestelmän avulla, mutta valmiiden töiden ja erääntyvien kuukausimaksujen vienti laskutukseen on puoliautomatisoitu. Projektinhallintaohjelman valmiit työt siirtyvät laskutustyökalun avulla csv-tiedostoon, joka lähetetään laskutusohjelmaan. Laskuihin liittyviä asiakas- ja tuotenimitietoja hallinnoidaan projektinhallintajärjestelmässä ja tiedot siirtyvät csv-tiedoston mukana. Näin vältytään samojen tietojen ylläpidolta kahdessa eri järjestelmässä.

5.8.4 Asiakkaiden hallinta

Projekteja varten asiakkaista tarvittava tieto on suhteellisen vähäistä. Kuitenkin, koska tietojen monistamiselta ollaan haluttu välttyä, ollaan samaan järjestelmään haluttu integroida asiakkaiden hallinta. Asiakkaiden hallintaa käyttävät ensisijaisesti myyjät. Myyjiä varten järjestelmästä löytyvät laskutettavien asiakkaiden lisäksi myös kaikki potentiaaliset asiakkaat. Asiakkaalle mahdolliset luokat ovat laskutettava, potentiaalinen, käynnissä, odottaa, hävitty ja vinkki. Tilan ja muiden perustietojen lisäksi asiakkaan ominaisuuksiin kuuluvat esimerkiksi toimiala, liikevaihto, yhteyshenkilön tiedot ja vapaamuotoinen logi yhteydenotoista. Asiakkaaseen liittyviä tietoja on todella paljon, sillä kaikilla järjestelmän käyttäjillä on asiakkaan käsitteeseen eri näkökulma. Pelkästään laskutustietoja on jo kymmeniä rivejä.

5.8.5 Vaatimusten ja dokumenttien hallinta

Vaatimusten ja dokumenttien hallintaa ei järjestelmään suunnitelmista huolimatta päädytty toteuttamaan. Toistaiseksi tietojen hallinta on edelleen hyvin vapaamuotoista. Hallinnointi tapahtuu tyypillisesti yrityksen sisäisellä verkkolevyllä tiedostojärjestelmässä. Yhteydenpito asiakkaaseen ja muu keskustelu voidaan käydä muilla tavoin.

Uutena työkaluna projektien yhteisöllisyyden tukemiseen on otettu koekäyttöön Basecamp [37signals, LLC, 2013]. Siinä ei ole kaikkia taloushallinnon ja hallituksen vaatimia työkaluja, eikä edes tuotannon seurantaan välttämättömiä ominaisuuksia, mutta siinä on kuitenkin hyvät yhteisölliset mahdollisuudet. Basecampia käytetäänkin uuden projektinhallintajärjestelmän tukena ja enemmänkin sähköpostin korvikkeena. Työkalua on käytetty vasta muutamassa projektissa, joten on vielä aikaista sanoa, säilyykö se osana yrityksen projektinhallintaa.

5.9 Uuden järjestelmän käyttöönotto

Uuden järjestelmän käyttöönotto oli vaivatonta. Käyttöönottoa helpotti merkittävästi se, että osa henkilöstöstä oli aktiivisesti kehitystyössä mukana. Käytännössä käyttöönottoon ei mennyt montaa tuntia. Käyttäjien perehdytyskään ei vienyt liiemmin aikaa tuotannon pienen koon, henkilöstön yleisen järjestelmäosaamisen ja avokonttorin tuomien etujen ansiosta. Lisäksi projektinhallintajärjestelmän kehityksessä mukana olleet käyttäjät opastivat muuta henkilöstöä aina tarpeen mukaan, joten seisokkeja käytössä ei tullut.

Laskutus tehtiin kerran kuukaudessa vielä ennen uuteen järjestelmään siirtymistä. Kuukausittaisen laskutustapahtuman jälkeen uudet työt syötettiin nyt uuteen järjestelmään. Koska tarve uudelle ratkaisulle oli akuutti, uusi järjestelmä otettiin käyttöön tärkeimpien toiminnallisuuksien valmistuttua. Seurantamahdollisuuksien toteutus ja muut kehitystyöt lähtivät käyntiin pian käyttöönoton jälkeen.

5.10 Käyttäjien tyytyväisyys uuteen projektinhallintasovellukseen

Esitin kaksi kuukautta sovelluksen käyttöönoton jälkeen yrityksen työntekijöille ja johdolle seurantakyselyn järjestelmän onnistumisesta verrattuna alkuvaatimuksiin. Kysely muodostui käyttäjien alkuperäisistä vaatimuksista ja vastaajan tuli arvioida vaatimuksen täyttymistä valitsemalla yksi neljästä tyytyväisyyttä kuvaavasta vaihtoehdosta. Kyselyn tulokset on ryhmitelty taulukoihin käyttäjäryhmien mukaan. Jokaisen rivin solujen arvojen määrä kertoo vastaajien kokonaismäärän.

Vaatimus	Puuttuu täysin	Puutteellinen	Hyvä	Erinomainen
Tulostavoitteiden seuranta		2	1	1
Tehtyjen tarjousten määrän seuranta		1	2	1
Toteutettujen projektien määrä ja laajuus			2	2
Myynnin jakauma tulolähteittäin	1	1	2	

Kuva 8 Hallituksen vaatimusten täyttyminen.

Hallituksen oli aikaisemmin mahdotonta seurata yrityksen tulostavoitteiden täyttymistä. Kuvasta 8 voidaan päätellä tämän tarpeen nyt täyttyneen tyydyttävästi. Osa hallituksen jäsenistä näki tulostavoitteiden seurannan edelleen vaatimattomaksi, sillä projektinhallintasovelluksesta näkee vain tulevien kuukausien tuoton

Vaatimus	Puuttuu täysin	Puutteellinen	Hyvä	Erinomainen
Sovelluksen ylläpidettävyys		1		
Sovelluksen muokattavuus			1	
Dokumentoinnin kattavuus	1			
Virheiden seuranta ja jäljitettävyys				1

Kuva 9 Ylläpitäjän vaatimusten täyttyminen.

ennusteen. Myös muille talouden seurannan mittareille olisi tarvetta. Lisäksi kaivattaisiin edelleen tarkemmin raportteja myyntituoton lähteistä.

Kuvasta 9 voidaan päätellä, että sovelluksen tekninen ylläpito on ainakin toistaiseksi vielä epämääräistä. Ylläpidettävyyteen vaikuttaa lähdekoodin laatu, johdonmukaisuus ja luettavuus. Samoin dokumentointi puuttuu täysin. Dokumentoinnin puuttuessa projektinhallintasovelluksen toteuttaja on käytännössä ainoa, joka pystyy tekemään suuria muutoksia lähdekoodiin. Puutteet ylläpidettävyydessä ovat seurausta kiireestä ja muiden rinnakkaisten töiden korkeammasta prioriteetista. Ylläpidettävyyden puutteista huolimatta virheiden jäljittäminen on ohjelmassa helppoa, kun loppukäyttäjät ovat suurimman osan ajasta heti tavoitettavissa. Näin virheiden luonne saadaan heti selville ja ongelma voidaan paikantaa.

Vaatimus	Puuttuu täysin	Puutteelliner	Hyvä	Erinomainen
Töiden siirto laskutusohjelmaan				1
Laskutettujen töiden seuranta		1		
Kuukausimaksullisten palveluiden laskutus				1
Kuluvan kuukauden laskutusennusteen hahmottaminen				1

Kuva 10 Taloushallinnon vaatimusten täyttyminen.

Taloushallinnon tarpeet onnistuttiin täyttämään erinomaisesti. Kuvassa 10 on esitetty taloushallinnon vaatimukset ja järjestelmätoteutuksen onnistuminen niiden ratkaisemisessa. Taloushallinto oli kaikin puolin tyytyväinen projektinhallintasovelluksen tuomiin laskutusta tukeviin ominaisuuksiin. Jatkokehityksenä voitaisiin vielä enemmän tuoda laskujen seurantamahdollisuuksia projektinhallintasovellukseen, jolloin voitaisiin vähentää kahden ohjelman käyttöä.

Kuten kuvan 11 taulukosta voidaan päätellä, tuotannon alkuperäiset tarpeet ja vaatimukset onnistuttiin täyttämään suurilta osin. Työtehtävien ja tuntikir-

Vaatimus	Puuttuu täysin	Puutteellinen	Hyvä	Erinomainen
Oman työkuorman kokonaisuuden hallinta		1	4	3
Omien työosioiden järjestyksen selkeys		1	3	4
Omien työosioiden aikatauluttaminen		3	3	2
Työtuntien kirjaamisen helppous		4	4	
Työtehtävien vaatimusten hallinta	2	4	2	

Kuva 11 Tuotannon työntekijöiden vaatimusten täyttyminen.

janpidon ylläpito pystytään nyt tekemään keskitetysti saman ohjelman avulla. Huomiota herätti vaatimustenhallinnan mahdollisuudet, johon saatiin vaihtelevia vastauksia. Käytännössä vaatimustenhallintaa ei oltu toteutettu, mutta jotkut käyttäjät alkoivat syöttää vaatimuksia työosioiden kuvaustekstin paikalle.

Vaatimus	Puuttuu täysin	Puutteellinen	Hyvä	Erinomainen
Asiakkaiden hallinta		3	1	
Tarjousten kirjaaminen		1	2	1
Tarjousten seuranta			2	2
Asiakkaalle tehtyjen projektien haku	1	1	2	

Kuva 12 Myynnin vaatimusten täyttyminen.

Kuvassa 12 on esitetty myyjien vaatimukset ja järjestelmätoteutuksen onnistuminen niiden ratkaisemisessa. Myynnin tukeminen sovelluksessa vaatii vastausten perusteella vielä jatkokehitystä. Ongelmana oli aikaisemmin asiakastietojen hajanaisuus myyjien omissa tiedostoissa. Nyt keskitetty ylläpito on mahdollista, mutta käytettävyys vaatii vielä kehitystä. Myyjien omat asiakasrekisterit exceltiedostoissa palvelivat aikaisemin paremmin jokaisen omia merkintätapoja, joten siirtyminen uuden sovelluksen käyttöön hidasti hieman työskentelyä.

5.11 Tulokset ja havainnot

Uuden järjestelmän käyttöönoton jälkeen, tehtiin teemahaastatteluja sen käyttäjille. Teemahaastattelu on vapaamuotoinen. Siinä haastateltava kertoo mielipiteistään liittyen ennalta määriteltyyn aiheeseen. Haastattelija voi tehdä haastattelun aikana muutoksia suunnittelemaansa kysymysrunkoon ja etenemisjärjestykseen haastateltavan antamien vastausten mukaisesti [Saaranen-Kauppinen & Puusniekka, 2013].

Tutkimusta varten haastateltiin seitsemää Mediasignal Communications Oy:n työntekijää. Kaikki haastattelut tehtiin yksitellen henkilökohtaisesti. Haastattelun alussa kysymykset liittyivät haastateltavan henkilön toimenkuvaan, jonka jälkeen siirryttiin keskustelemaan uudesta projektinhallintajärjestelmästä. Haastateltavien työnkuvat olivat toimitusjohtaja (1), myyjä (1), web-ohjelmoija (2), graafikko (2) ja laskuttaja (1).

Työnkuvasta keskustelun jälkeen toisena kysymyksenä kysyin käyttäjän aikaisemmista kokemuksista projektinhallintajärjestelmien käytöstä. Suurin osa ei ollut käyttänyt aikaisemmin mitään, tai oli käyttänyt exceliä. Yksi henkilö oli käyttänyt SAP-toiminnanohjausjärjestelmää.

Seuraavaksi edettiin jo uuden projektinhallintajärjestelmän käyttöön ja pyysin henkilöä kertomaan päivittäin käyttämistään ominaisuuksista. Vastaukset osuivat hyvin yhteen projektinhallintajärjestelmän suunnitellun käytön ja etukäteen tunnistettujen roolien kanssa. Pienenä roolien ristiriitana oli vanhojen töiden haku ja seuranta, jota tehdään laskuttajalle suunnitellusta osiosta laskujen historiatietojen kautta.

Uuden projektinhallintajärjestelmän koettiin yleisesti helpottaneen työntekoa. Laskuttamiseen kuluva aika on nykyään murto-osa aikaisemmasta. Aikaisemmin laskut kirjattiin uuteen järjestelmään käsin excel-taulukosta kopioimalla. Nykyään laskut siirtyvät projektinhallintajärjestelmästä laskutusohjelmaan muutamalla napin painalluksella. Projektien ja työosioiden merkitseminen vaatii suunnilleen saman verran työtä kuin aikaisemminkin excel-taulukoita käytettäessä. Töiden seuranta ja tuotantotilanteen ennustaminen on nyt kuitenkin mahdollista, kun ennen se oli mahdotonta.

Projekteilla ja työosioilla on niin paljon laskutuksen ja seurannan kannalta oleellista tietoa, että se hidastaa ja hankaloittaa tuotannon työntekijöiden työntekoa hieman. Laskutussuunnitelma toi lähes jokaiselle ongelmia. Se mahdollistaa projektien laskuttamisen useassa osassa. Ominaisuus on tarkoitettu isommille projekteille ja se aiheuttaa hämmennystä esimerkiksi päivitystöitä tehdessä. Muitakin ongelmia löytyi. Eräs ongelma oli historiatietojen seuraaminen, esimerkiksi web-ohjelmoijan omien jo tehtyjen töiden seuranta. Historiatietoja on mahdollista hakea laskutushistorian kautta, mutta olisi mukavampi jos työt näkisi jostain helpommin. Tällä hetkellä jokaisella käyttäjällä on sovelluksessa oma sivu, josta

näkee keskeneräiset ja tulevat työt. Toinen useampaan kertaan mainittu ongelma oli projektien toteutuneiden tuntitietojen seuranta. Projekteille on mahdollista syöttää erikseen arvioidut ja toteutuneet tunnit. Toteutuneiden tuntien syöttö ei kuitenkaan ole niin helppoa, että sitä tulisi aina tehtyä. Haastattelun perusteella ongelma on käyttöliittymässä.

Laskuttaja suosittelisi järjestelmää muillekin yrityksille, jotka käyttävät samaa laskutusohjelmaa. Myös ohjelmoijat ja graafikot saattaisivat suositella vastaavaa projektinhallintajärjestelmää. Toimitusjohtajalla oli hieman ristiriitaisempi mielipide. Toimitusjohtajan näkökulmasta yksi puutteista on projektien tuntitietojen yhdistäminen tiettyyn henkilöön. Tällä hetkellä tehdyt tunnit merkataan samaan tuntisummaan projektin tietoihin. Toimitusjohtajalle olisi kuitenkin tärkeää pystyä seuraamaan työntekijöiden tuntikuormaa, jolloin töiden jakaminen ja tasapainottaminen olisi helpompaa. Käyttöliittymän puutteita on enemmän. Käyttöliittymän toteutus on kuitenkin vielä kesken ja saadun palautteen avulla siitäkin saadaan varmasti vielä tarkoitukseen sopiva.

Järjestelmän kehitys ja mukautuvuus olisi varmasti hankalampaa, mikäli projekti oltaisiin toteutettu ulkoisena työnä. Nyt kun projektinhallintajärjestelmän kehittäjä, käyttäjät ja asiakas ovat kaikki saman organisaation työntekijöitä, on järjestelmään mahdollista tehdä muutoksia nopealla aikataululla.

6 Yhteenveto

Valmiin ratkaisun löytäminen ohjelmistoprojektin hallitsemiseksi ei osoittautunut mahdolliseksi Mediasignal Groupin tapauksessa, vaikka vaihtoehtoja kartoitettiin tässä prosessissa runsaasti ja potentiaalisimpia vaihtoehtoja testattiin käytännössä. Näin oli siitä huolimatta, että kohdeyritys toimii IT-alalla ja henkilöstöllä on keskimääräistä paremmat valmiudet uusien järjestelmien käyttöönottoon. Valmiin ratkaisun käyttöönotto olisi edellyttänyt yrityksen toimintatapojen muuttamista ohjelmaan sopivaksi. Suurin osa tutkituista tuotteista oli joko liian kevyitä ratkaisuja Mediasignal Groupin käyttötarkoituksiin tai käytettävyydeltään vaatimustasoa heikompia.

Toisaalta, jos kaikki valmiit ratkaisut vaikuttavat huonoilta tai riittämättömiltä, herää kysymys yrittävätkö ohjelmistojen valmistajat kohdistaa yhden ja saman ratkaisun liian laajalle kohderyhmälle. Yritysten toimintatapojen monipuolista kirjoa ei tutkituissa ratkaisuissa oltu huomioitu kovinkaan hyvin. Edes se, mille toimialalle ratkaisu on ensisijaisesti suuniteltu, ei yleensä tullut esiin. Johtopäätös tästä on, että projektinhallintaohjelmistoja kehittävien tahojen keskuudessa asiakaslähtöisessä ajattelussa on vielä paljon enemmän kehitettävää kuin monissa muussa IT-alan ratkaisuissa. Ajateltaessa projektinhallintaohjelmistoja hankkivia yrityksiä niiden suosiota vasten herää kysymys, yritetäänkö projektien vajavaista hallintaa ja seurantaa tai projektipäällikön johtamistaitoja korjata hankkimalla projektinhallintajärjestelmä, jonka projektinhallintaa tehostavaan vaikutukseen uskotaan sokeasti. Selvää lienee, että projektinhallintajärjestelmän tarkoituksena ei ole hallita projektia, eikä yksin sen voi olettaa parantavan merkittävästi heikkoja ryhmätyötaitoja.

Todennäköisesti myös talousresurssien rajallisuus ajaa pk-yrityksiä hankkimaan usein valmisohjelmiston oman ohjelmiston kehittämisen sijaa. On selvää, että valmisohjelmisto on kertainvestointina selvästi oman sovelluksen kehitystä edullisempi tai open source –sovelluksena ilmainen. Tästä syntyy kuitenkin helposti suuret kustannukset, jos sovellus ei tehosta projektien hallintaa vaan hankaloittaa sitä. Näin kävi myös Mediasignal Communications Oy:n hankkiessa ensin valmiin kaupallisen ohjelmiston. Koulutukseen ja uuden käytännön lanseeraukseen käytettiin paljon resursseja, mutta seurauksena oli palaaminen vanhaan menettelyyn.

Yleisesti on todettava, että projektinhallinnan tehostaminen ei liene koskaan

helppo tehtävä, eikä ohjelmiston hankinta sitä itsestäänselvästi ainakaan helpota. Siten kevyiden ratkaisujen suuri määrä ja suosio tuntuu aivan liialliselta.

Myös sovellusten valmistajien ja myyjien vastuu herättää ajatuksia. Sovellusten esittäminen yleispätevästi yritysten projekteja tehostavina ratkaisuina on vähintäänkin arveluttavaa. Usein sovellusten markkinoinnista saa tällaisen kuvan. Tämä koskee monesti myös laajempia sovelluksia.

Joka tapauksessa on selvää, että oletus projektinhallintajärjestelmän hankinnasta ilman jatkokehityksen tarpeita on ehdottoman väärä. Jotta projektinhallintajärjestelmä voisi toimia kasvuyrityksen tukena, sen kehittämisen on jatkuttava koko yrityksen elinkaaren ajan, yrityksen toiminnan kehittymisen tahdissa.

Viiteluettelo

[37signals, LLC, 2013] 37signals, LLC. Basecamp, 2013. `https://basecamp.com/` [Accessed May 25, 2013].

[A51 doo, 2013] A51 doo. activecollab, 2013. `https://www.activecollab.com` [Accesssed May 25, 2013].

[Abran & Moore, 2004] Alain Abran & James W. Moore. Chapter 2: Software requirements. In *Guide to the Software Engineering Body of Knowledge*, 2004.

[Adobe, 2013] Adobe. Action method, 2013. `https://www.actionmethod.com` [Accesssed May 25, 2013].

[Babar, 2009] Muhammed Ali Babar. Design decisions and design rationale in software architecture. *The Journal of Systems and Software*, 82:1195–1197, 2009.

[Badia, 2011] Antonio Badia. A call to arms: revisiting database design. *SIGMOD Record Volume 40 Issue 3*, 2011.

[Bass *et al.*, 2012] Len Bass, Paul Clements, & Rick Kazman. *Software Architecture in Practice*. Addison-Wesley, 2012.

[Boehm, 1989] Barry W. Boehm. *Software Risk Management*. IEEE Computer Society Press, 1989.

[Bosch, 1997] Weck Bosch, Szyperski. Summary of the second international workshop on component-oriented programming. *International Workshop on Component-Oriented Programming*, pages 94–101, 1997.

[Chatfield, 2007] Johnson Chatfield. *A Short Course in Project Management*. Microsoft, 2007.

[Christel & Kang, 1992] Michael Christel & Kyo C. Kang. Issues in requirements elicitation. Technical report, 1992.

[Cotterell & Hughes, 2009] Mike Cotterell & Bob Hughes. *Software Project Management*. McGraw-Hill Education, 2009.

[Delgadillo & Gotel, 2007] Lorena Delgadillo & O Gotel. Story-wall: A concept for lightweight requirements management. In *15th IEEE International Requirements Engineering Conference*, pages 377–378, 2007.

[Dorota Huizinga, 2007] Adam Kolawa Dorota Huizinga. *Automated Defect Prevention: Best Practices in Software Management*. Wiley-IEEE, 2007.

[Dudziak, 2000] Thomas Dudziak. Extreme programming an overview. *Methoden und Werkzeuge der Softwareproduktion WS*, 2000.

[Gauthier, 2013] Alexandre Gauthier. What is Scrum, 2013. `https://www.planbox.com/resources/agile-artifacts#roles` [Accesssed May 14, 2013].

[Hughes, 2006] Cotterell M. Hughes, B. *Project Management*. McGraw-Hill Education, 2006.

[IEEE, 1990] Computer Society IEEE. Ieee standard glossary of software engineering terminology, 1990.

[ILX Group, 2013a] ILX Group. Prince2 - a structured project management methodology, 2013. `http://www.prince2.com/prince2-methodology.asp#prince2-project-management-roles` [Accesssed May 14, 2013].

[ILX Group, 2013b] ILX Group. What is Prince2, 2013. `http://www.prince2.com/what-is-prince2.asp` [Accesssed May 14, 2013].

[Jama Software, 2011] Jama Software. Requirements management 101, 2011. `http://www.jamasoftware.com/` [Accesssed May 13, 2013].

[Kontio, 2005] Mikko Kontio. Designing software architectures. *Architectural manifesto*, 2005.

[Kotonya & Sommerville, 1998] Gerald Kotonya & Ian Sommerville. *Requirements Engineering*. John Wiley and Sons, 1998.

[Kruchten, 1995] Philippe Kruchten. Architectural blueprints — the "4+1" view model of software architecture. *IEEE Software 12 (6)*, pages 42–50, 1995.

[Kruchten, 2008] Philippe Kruchten. What do software architects really do. *Journal of Systems and Software*, page 2413–2416, 2008.

[Kwak, 2003] Young Hoon Kwak. Brief history of project management. In *The Story of Managing Projects*. Quorum Books, 2003.

[Lang, 2013] Jean-Philippe Lang. Redmine, 2013. `http://www.redmine.org/` [Accesssed May 25, 2013].

[Lowell Lindstrom, 2003] Ron Jeffries Lowell Lindstrom. Extreme programming and agile software development methodologies. *CRC Press LLC*, 2003.

[Mediasignal Communications Oy, 2013] Mediasignal Communications Oy. Mediasignal Communicationsin projektiarkisto, 2013.

[Pivotal Labs, 2013] Inc. Pivotal Labs. Pivotal tracker, 2013. `https://www.pivotaltracker.com` [Accesssed May 25, 2013].

[Project Management Institute, 2009] Project Management Institute. *A Guide to the Project Management Body of Knowledge*. Project Management Institute, 2009.

[Romano *et al.*, 2004] Nicholas Romano, Fang Chen, & Jay Nunamaker. Chapter 4: Software construction. In *Guide to the Software Engineering Body of Knowledge*, 2004.

[Rowel & Alfeche, 1997] Ramos Rowel & Kurts Alfeche. *Requirements Engineering A Good Practice Guide*. John Wiley and Sons, 1997.

[Saaranen-Kauppinen & Puusniekka, 2013] Anita Saaranen-Kauppinen & Anna Puusniekka. Teemahaastattelu, 2013. `http://www.fsd.uta.fi/menetelmaopetus/kvali/L6_3_2.html` [Accesssed May 25, 2013].

[Sapolsky, 2004] Harvey Sapolsky. The U.S. Navy's Fleet Ballistic Missile Program And Finite Deterrence. 2004.

[SARA Work Group, 2002] SARA Work Group. Sara report, 2002.

[Schwaber & Sutherland, 2013] Ken Schwaber & Jeff Sutherland. Thescrumguide. *Scrum.org*, 2013.

[Smith, 2008] Rachel S. Smith. Writing a requirements document. *CSU Center for Distributed Learning*, 2008.

[Sommerville, 2011] Ian Sommerville. *Software Engineering.* Addison-Wesley Computing, 2011.

[StoryWall, 2011] StoryWall. Storywall, 2011. `http://storywallapp.com/` [Accesssed June 4, 2013].

[Tran & Liu, 1999] Vu Tran & Dar-Biau Liu. A procurement-centric model for engineering component-based software systems. In *The Twenty-Second Annual International SIGIR Conference on Research and Development in Information Retrieval*, pages 70–79, 1999.